Modbus Router / Modbus Router 485

User Manual

A-MBR / A-MBR-485

Document No. D106-009
09/2015
Revision 1.5

aparian

CONTENTS

Revision History

Revision	Date	Comment
1.0	24 June 2015	Initial document
1.1	24 July 2015	Indication of serial data and stop bits supported
1.2	25 August 2015	Add UL Listed mark
1.3	7 September 2015	Added documentation for the RS485 version
1.4	28 September 2015	Added section for Base Address selection
1.5	30 September 2015	Add support for EtherNet/IP retry statistics and configuration

1. PREFACE

1.1. INTRODUCTION TO THE MODBUS ROUTER

This manual describes the installation, operation, and diagnostics of the Aparian Modbus Router. The Modbus Router provides intelligent data routing between EtherNet/IP and Modbus (serial Modbus-RTU or Ethernet Modbus-TCP). **NOTE:** The Modbus Router 485 can communicate on serial RS485 whilst the Modbus Router can communicate on serial RS232. The Modbus Router allows the user to integrate Modbus devices into a Rockwell Logix platform (e.g. ControlLogix or CompactLogix) with minimal effort.

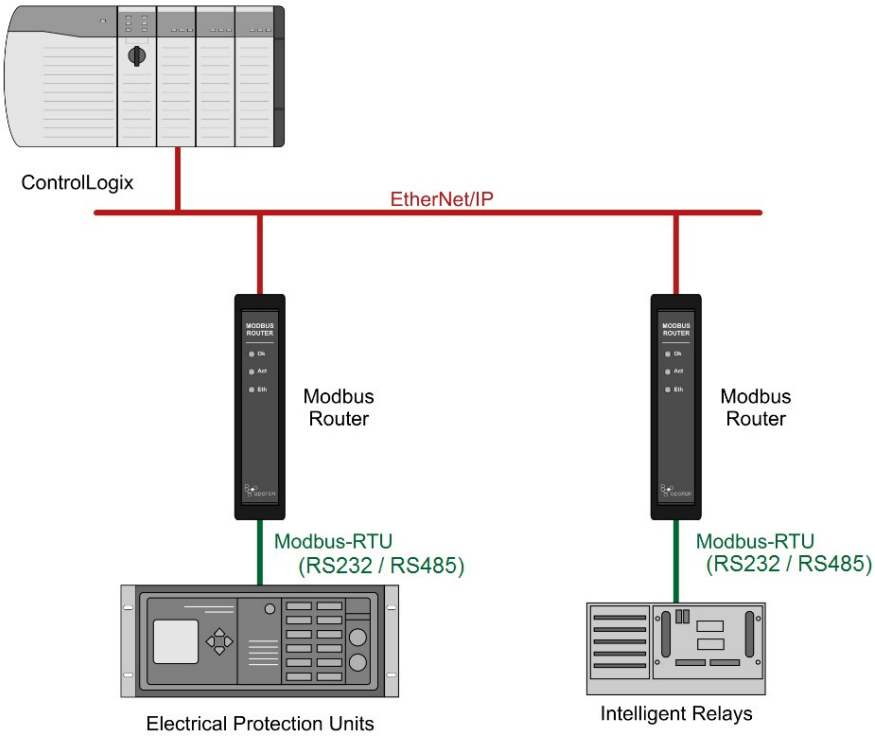

Figure 1.1. – Typical Setup

1.2. FEATURES

The Modbus Router is able to transfer data from various Modbus devices to a maximum of three Logix controllers. The module operates in one of three modes, simplifying the configuration for all applications.

Mode	Description	Message Initiator
Reactive Tag	The Modbus Router will convert Modbus messages to Logix controller tag reads or tag writes. (Modbus Slave)	Remote Device

Scheduled Tag	The Modbus Router transfers data between a Modbus device and a number of Logix tags, using a preconfigured scheduled. (Modbus Master) No Logix or remote device configuration is required.	Modbus Router
Unscheduled	The Modbus Router transfers messages received from a Logix Message Instruction. (Modbus Master)	Logix (Msg)

Table 1.1. – Modes of Operation

The Modbus Router is configured using the Aparian Slate application. This program can be downloaded from www.aparian.com free of charge. Slate offers various configuration methods, including a controller tag browser.

Hereafter the Modbus Router will be referred to as the **module**.

The module can operate in both a Logix "owned" and standalone mode. With a Logix connection the input and output assemblies will provide additional diagnostics information which will be available in the Logix controller environment.

The Modbus Router allows the user to integrate Modbus devices into a Logix system with minimal effort. No copying or mapping of data in the Logix controller is required as the Modbus Router writes directly into Logix tags.

The module also provides a range of statistics and traffic analyser to help fault find any problems.

The Modbus Router supports Modbus on two ports which can be configured from the Slate environment; Modbus-RTU (Serial) or Modbus-TCP (Ethernet).

The Modbus Router (RS232 version) uses isolated RS232 for Modbus serial communication providing better noise immunity. The RS232 or RS485 port uses a terminal block for convenient installation.

A built-in webserver provides detailed diagnostics of system configuration and operation, including the display of Modbus operation and communication statistics, without the need for any additional software.

1.3. ARCHITECTURE

The figure below provides an example of the typical network setup in reactive mode, where the Modbus Router acts as a Modbus slave device.

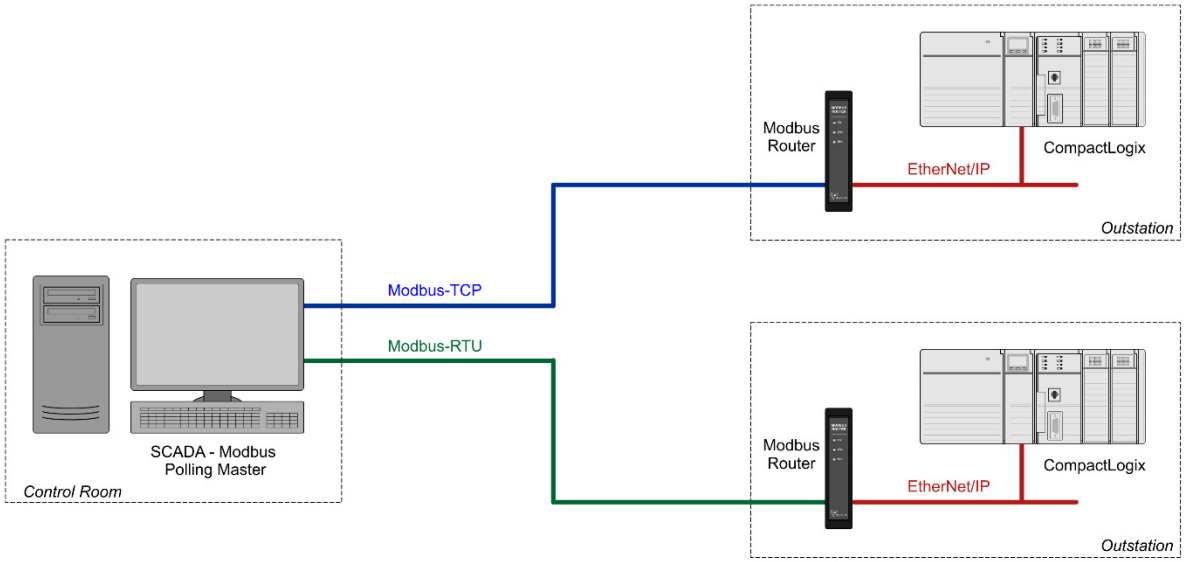

Figure 1.2. - Example of a typical network setup in reactive mode

By converting and redirecting serial Modbus messages from legacy devices to EtherNet/IP, the module provides an interface for data exchange to Allen-Bradley ControlLogix and CompactLogix platforms. This enables user to replace legacy devices and systems with minimal effort and downtime.

The Modbus Router allows a Logix platform to seamlessly integrate into a Modbus network with Reactive Tag Mode. The module will route Modbus message directly to Logix tags with no need for additional ladder code.

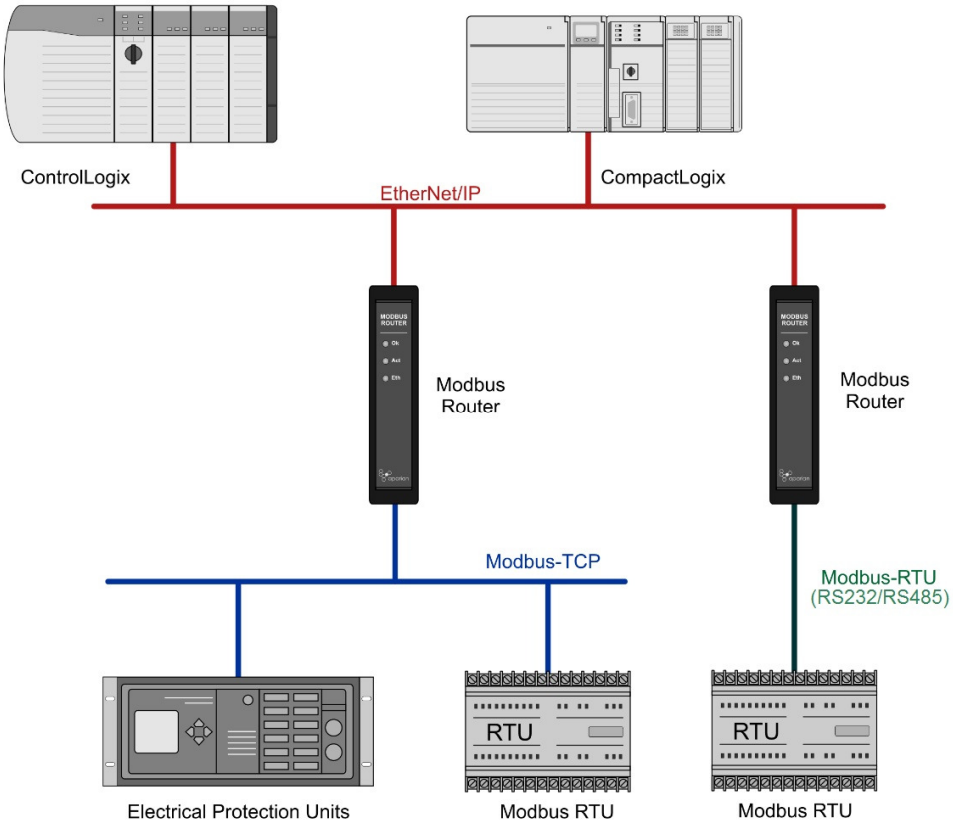

Figure 1.3. - Example of a typical network setup in scheduled/unscheduled mode

Systems that rely on a central ControlLogix communicating to a number of remote Modbus devices, (e.g. Electrical Protection Units or Remote Terminal Units), may find the Modbus Router useful when operating in Scheduled Tag Mode as shown in the figure above. The module, acting as a Modbus master, will exchange data between the Modbus device and Logix platform at a configured interval without any need for additional coding or mapping.

1.4. ADDITIONAL INFORMATION

The following documents contain additional information that can assist the user with the module installation and operation.

Resource	Link
Slate Installation	http://www.aparian.com/software/slate
Modbus Router User Manual Modbus Router Datasheet Example Code & UDTs	http://www.aparian.com/products/modbusrouter
Ethernet wiring standard	www.cisco.com/c/en/us/td/docs/video/cds/cde/cde205_220_420/installation/guide/cde205_220_420_hig/Connectors.html
CIP Routing	The CIP Networks Library, Volume 1, Appendix C:Data Management
Modbus	http://www.modbus.org

Table 1.2. - Additional Information

1.5. SUPPORT

Technical support is provided via the Web (in the form of user manuals, FAQ, datasheets etc.) to assist with installation, operation, and diagnostics.

For additional support the user can use either of the following:

Resource	Link
Contact Us web link	www.aparian.com/contact-us
Support email	support@aparian.com

Table 1.3. – Support Details

2. INSTALLATION

2.1. MODULE LAYOUT

The module has three ports at the bottom of the enclosure as shown in the figure below. The ports are used for Ethernet, RS232 or RS485 serial, and power. The power port uses a three way connector which is used for the DC power supply positive and negative (or ground) voltage as well as the earth connection.

The Ethernet cable must be wired according to industry standards which can be found in the additional information section of this document.

When using the RS232 version the RS232 port uses a four way connector. This provides connection for the communication transmit (TX), receive (RX), and ground (GND) conductors. The fourth connection (earth) is used for shielding the cable in high noise environments.

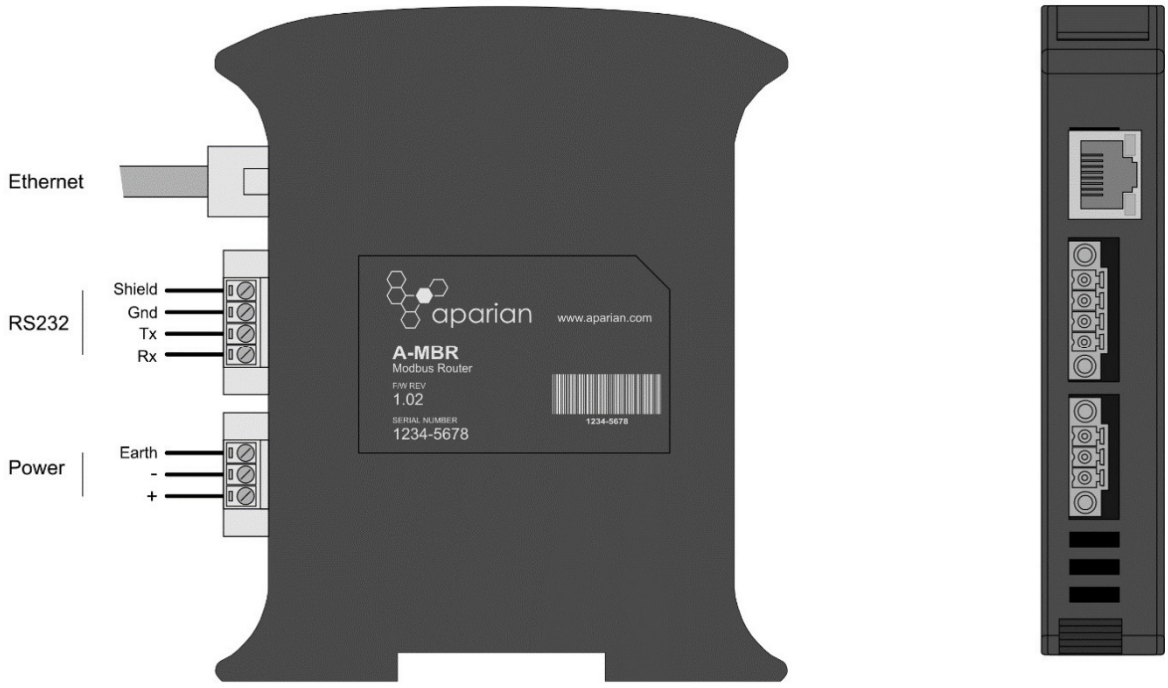

Figure 2.1. – Modbus Router (RS232) side and bottom view

When using the RS485 version the RS485 port uses a four way connector. This provides connection for the communication positive (+) and negative (-) conductors (the third connector is N/A). The fourth connection (earth) is used for shielding the cable in high noise environments.

Figure 2.2 – Modbus Router (RS485) side and bottom view

The module provides three diagnostic LEDs as shown in the front view figure below. These LEDs are used to provide information regarding the module system operation, the Ethernet interface, and the auxiliary communication interface (RS232 or RS485).

Figure 2.3. – Modbus Router front and top view

The module provides four DIP switches at the top of the enclosure as shown in the top view figure above.

DIP Switch	Description
DIP Switch 1	Used to force the module into “Safe Mode”. When in “Safe Mode” the module will not load the application firmware and will wait for new firmware to be downloaded. This should only be used in the rare occasion when a firmware update was interrupted at a critical stage.
DIP Switch 2	This will force the module into DHCP mode which is useful when the user has forgotten the IP address of the module.
DIP Switch 3	Reserved
DIP Switch 4	Reserved

Table 2.1. - DIP Switch Settings

2.2. MODULE MOUNTING

The module provides a DIN rail clip to mount onto a 35mm DIN rail.

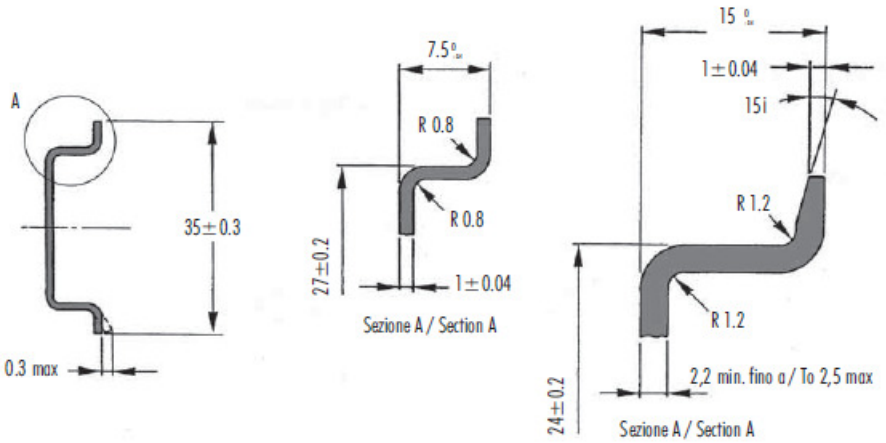

Figure 2.4 - DIN rail specification

The DIN rail clip is mounted on the bottom of the module at the back as shown in the figure below. Use a flat screw driver to pull the clip downward. This will enable the user to mount the module onto the DIN rail. Once the module is mounted onto the DIN rail the clip must be pushed upwards to lock the module onto the DIN rail.

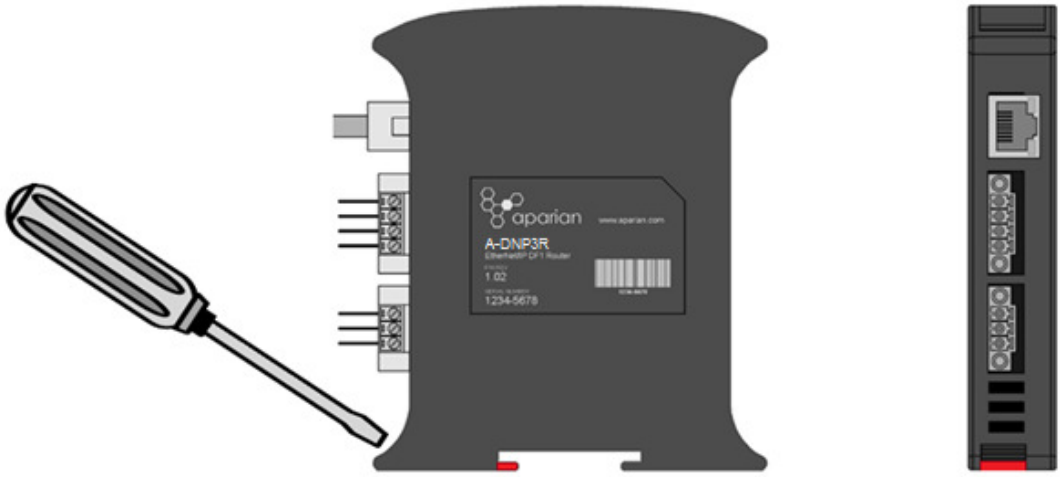

Figure 2.5 - DIN rail mouting

2.3. POWER

A three way power connector is used to connect Power+, Power– (GND), and earth. The module requires an input voltage of 10 – 28Vdc. **Refer** to the technical specifications section in this document.

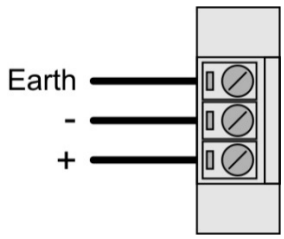

Figure 2.6 - Power connector

2.4. RS232 PORT

NOTE: The RS232 port is only applicable with the RS232 version of the Modbus Router (catalog A-MBR).

The four way RS232 connector is used to connect the transmit (TX), receive (RX), and GND conductors for serial communication. The shield terminal can be used for shielded cable in high noise environments.

NOTE: The shield of the RS232 port is internally connected to the power connector earth. Thus when using a shield it is important to connect the Earth terminal on the power connector to a clean earth. Failing to do this can lower the signal quality of the RS232 communication.

NOTE: When using a shielded cable, it is important that only one end of the shield is connected to earth to avoid current loops. It is recommended to connect the shield to the Modbus Router module, and not to the other Modbus device.

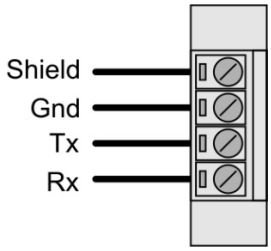

Figure 2.7 - RS232 connector

2.5. RS485 PORT

NOTE: The RS485 port is only applicable with the RS485 version of the Modbus Router (catalog A-MBR-485).

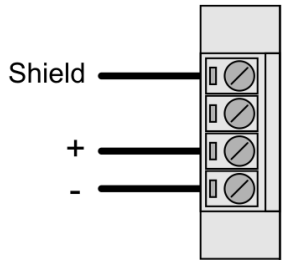

Figure 2.8 – RS485 connector

The four way RS485 connector is used to connect the positive (+) and negative (-) communication conductors for serial communication. The shield terminal can be used for shielded cable in high noise environments.

NOTE: The shield of the RS485 port is internally connected to the power connector earth. Thus when using a shield it is important to connect the Earth terminal on the power connector to a clean earth. Failing to do this can lower the signal quality of the RS485 communication.

NOTE: When using a shielded cable, it is important that only one end of the shield is connected to earth to avoid current loops. It is recommended to connect the shield to the Modbus Router module, and not to the other Modbus device.

2.6. RS485 TERMINATION

NOTE: The RS485 port is only applicable with the RS485 version of the Modbus Router (catalog A-MBR-485).

All RS485 networks need to be terminated at the extremities (start and end point) of the communication conductor. The termination is done by placing a resistor between the positive and negative communication conductor. The value of the resistor will depend on the characteristic impedance of the cable chosen, but generally ranges from 100 Ohm to 150 Ohm.

2.7. ETHERNET PORT

The Ethernet connector should be wired according to industry standards. **Refer** to the additional information section in this document for further details.

3. SETUP

3.1. INSTALL CONFIGURATION SOFTWARE

All the network setup and configuration of the module is achieved by means of the Aparian Slate device configuration environment. This software can be downloaded from http://www.aparian.com/software/slate.

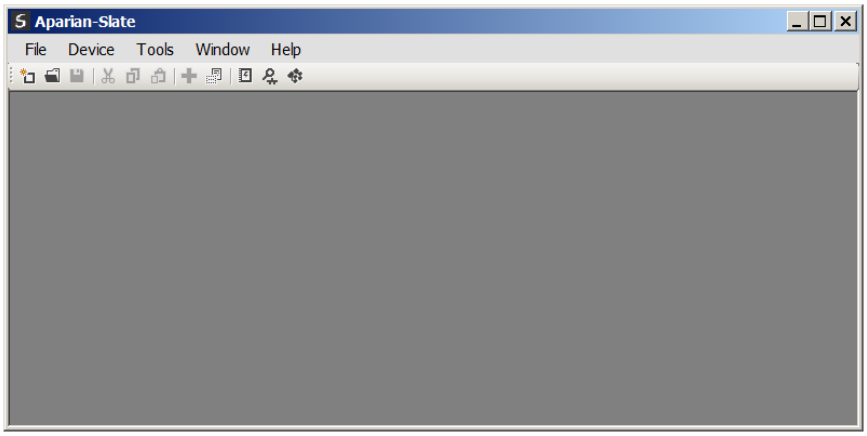

Figure 3.1. - Aparian Slate Environment

3.2. NETWORK PARAMETERS

The module will have DHCP (Dynamic Host Configuration Protocol) enabled as factory default. Thus a DHCP server must be used to provide the module with the required network parameters (IP address, subnet mask, etc.). There are a number of DHCP utilities available, however it is recommended that the DHCP server in Slate be used.

Within the Slate environment, the DHCP server can be found under the Tools menu.

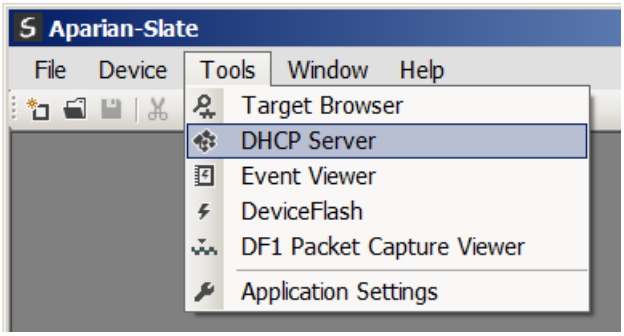

Figure 3.2. - Selecting DHCP Server

Once opened, the DHCP server will listen on all available network adapters for DHCP requests and display their corresponding MAC addresses.

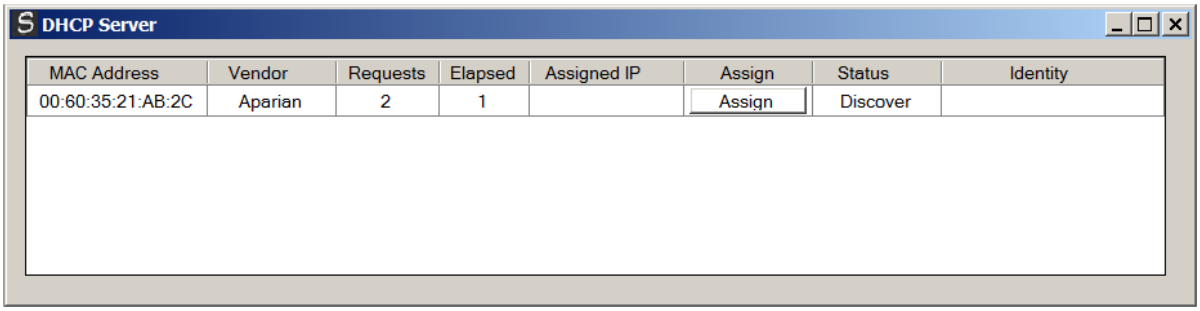

Figure 3.3. - DHCP Server

NOTE: If the DHCP requests are not displayed in the DHCP Server it may be due to the local PC's firewall. During installation the necessary firewall rules are automatically created for the Windows firewall. Another possibility is that another DHCP Server is operational on the network and it has assigned the IP address.

To assign an IP address, click on the corresponding "Assign" button. The IP Address Assignment window will open.

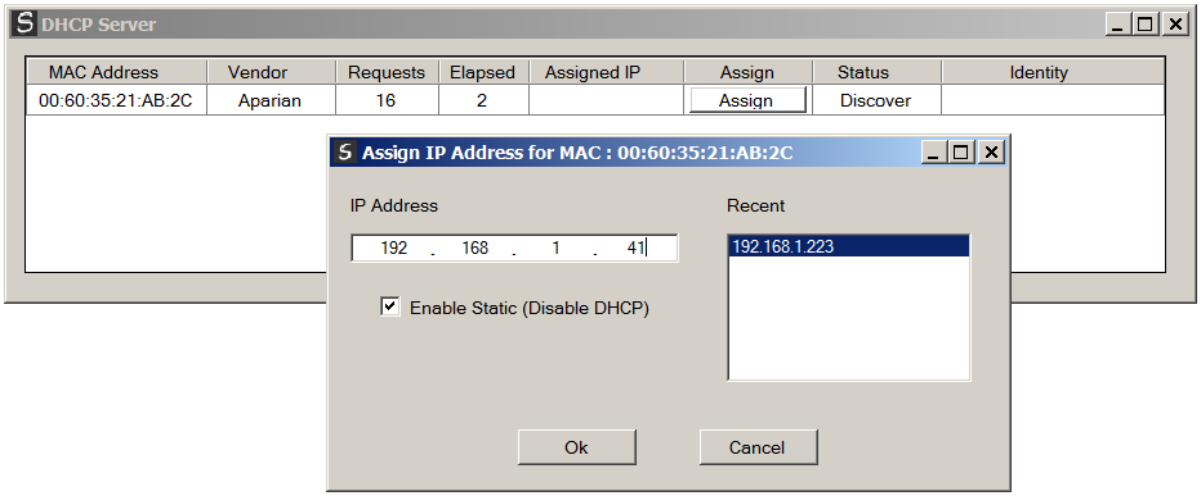

Figure 3.4. - Assigning IP Address

The required IP address can then be either entered, or a recently used IP address can be selected by clicking on an item in the Recent List.

If the “Enable Static” checkbox is checked, then the IP address will be set to static after the IP assignment, thereby disabling future DHCP requests.

Once the IP address window has been accepted, the DHCP server will automatically assign the IP address to the module and then read the Identity object Product name from the device.

The successful assignment of the IP address by the device is indicated by the green background of the associated row.

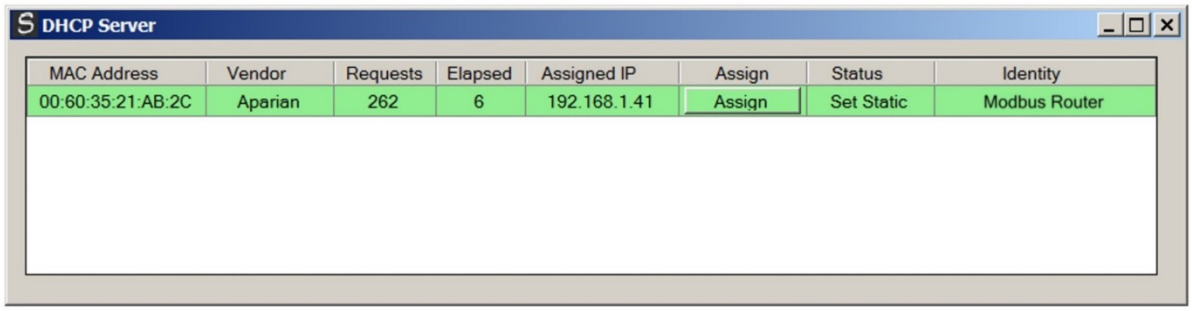

Figure 3.5. - Successful IP address assignment

It is possible to force the module back into DHCP mode by powering up the device with DIP switch 2 set to the On position.

A new IP address can then be assigned by repeating the previous steps.

NOTE: It is important to return DIP switch 2 back to Off position, to avoid the module returning to a DHCP mode after the power is cycled again.

If the module's DIP switch 2 is in the On position during the address assignment, the user will be warned by the following message.

Figure 3.6. - Force DHCP warning

In addition to the setting the IP address, a number of other network parameters can be set during the DHCP process. These settings can be viewed and edited in Slate's Application Settings, in the DHCP Server tab.

Once the DHCP process has been completed, the network settings can be set using the Ethernet Port Configuration via the Target Browser.

The Target Browser can be accessed under the Tools menu.

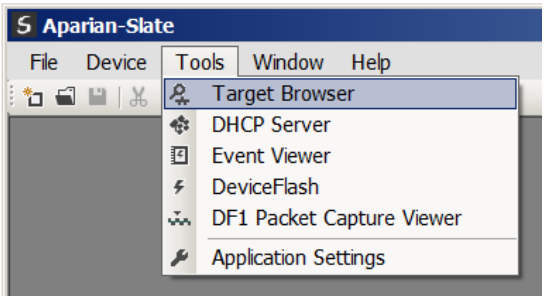

Figure 3.7. - Selecting the Target Browser

The Target Browser automatically scans the Ethernet network for EtherNet/IP devices.

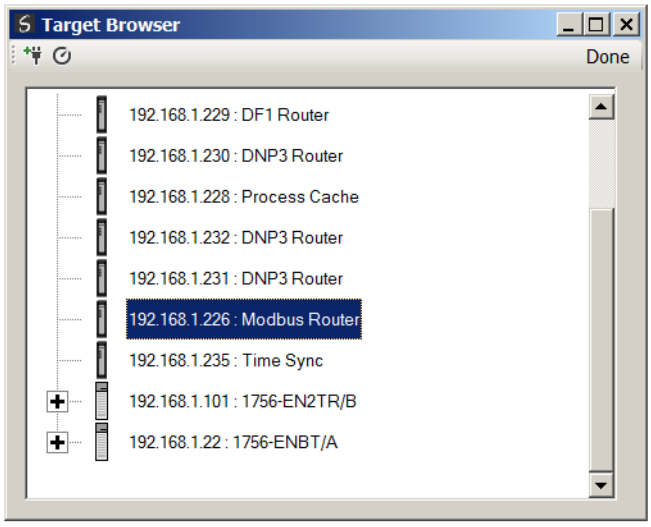

Figure 3.8. - Target Browser

Right-clicking on a device, reveals the context menu, including the Port Configuration option.

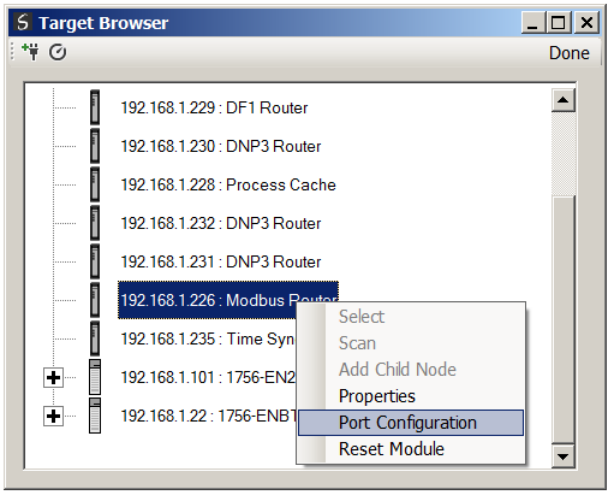

Figure 3.9. - Selecting Port Configuration

All the relevant Ethernet port configuration parameters can be modified using the Port Configuration window.

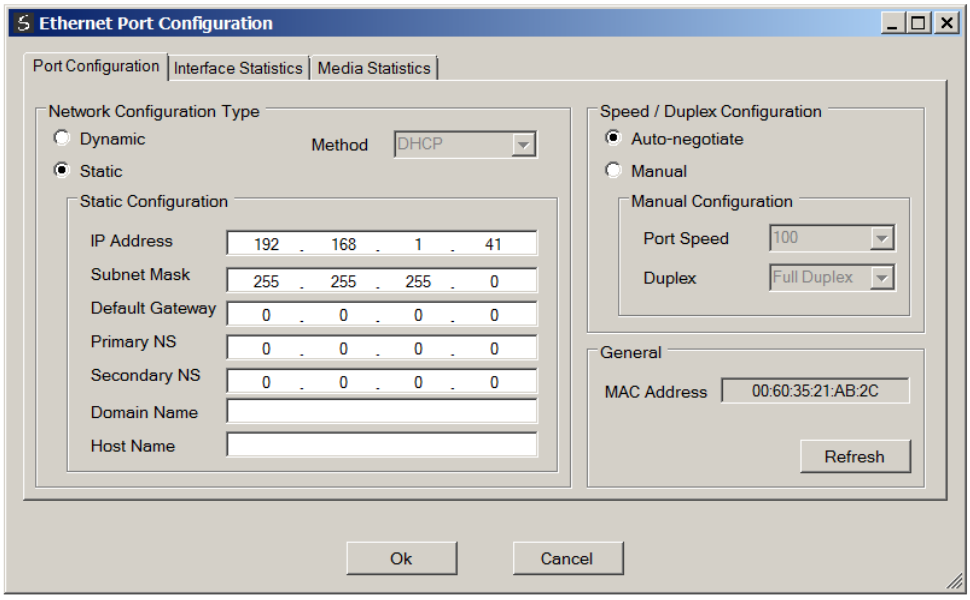

Figure 3.10. - Port Configuration

Alternatively, these parameters can be modified using Rockwell Automation's RSLinx software.

3.3. CREATING A NEW PROJECT

Before the user can configure the module, a new Slate project must be created. Under the File menu, select New.

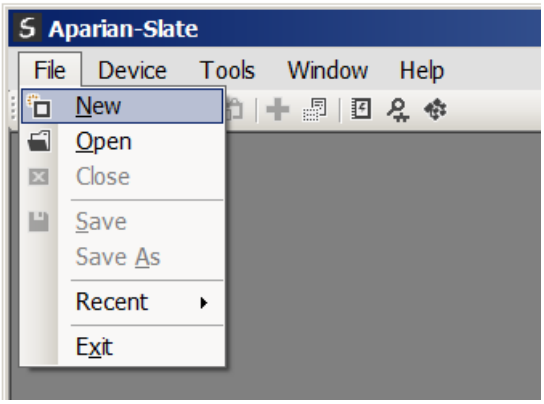

Figure 3.11. - Creating a new project

A Slate project will be created, showing the Project Explorer tree view. To save the project use the Save option under the File menu.

A new device can now be added by selecting Add under the Device menu.

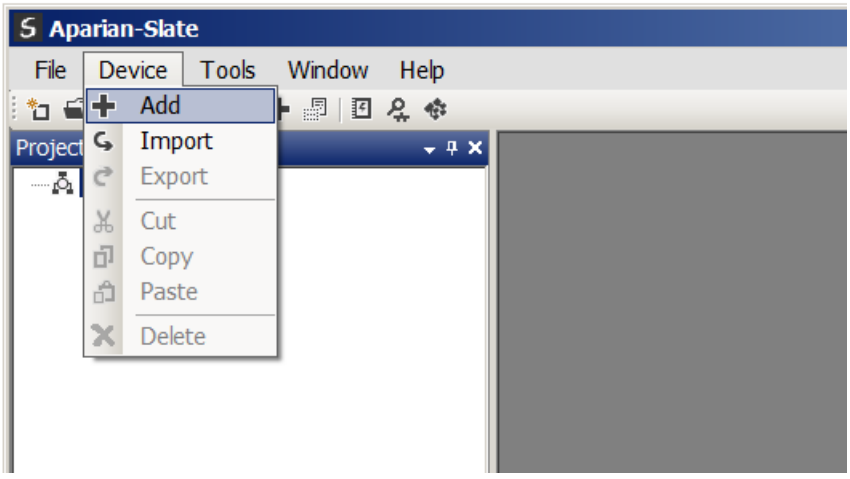

Figure 3.12. - Adding a new device

In the Add New Device window select the Modbus Router, and click the Ok button.

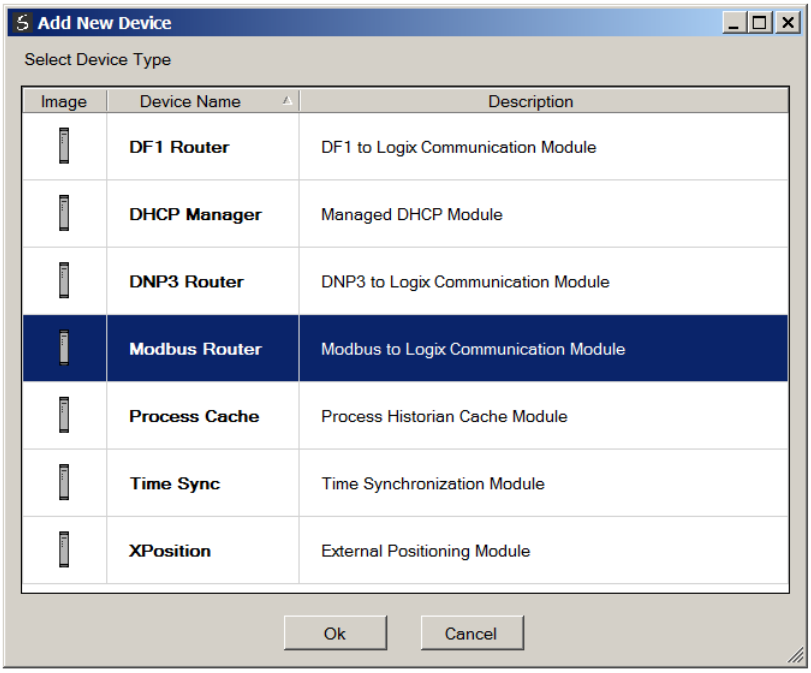

Figure 3.13 – Selecting a new Modbus Router

The device will appear in the Project Explorer tree as shown below, and its configuration window opened.

The device configuration window can be reopened by either double clicking the module in the Project Explorer tree or right-clicking the module and selecting *Configuration*.

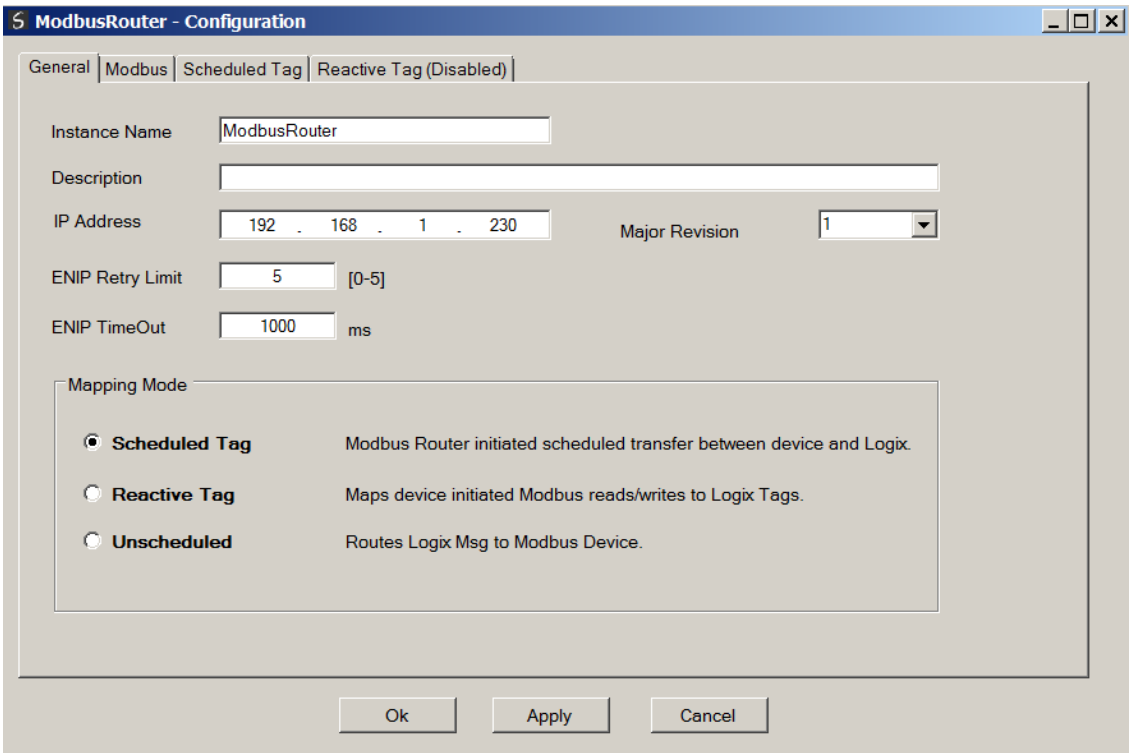

Figure 3.14. – Modbus Router configuration

Refer to the additional information section in this document for Slate's installation and operation documentation.

3.4. MODBUS PARAMETERS

The Modbus parameters will be configured by Slate. **Refer** to the additional information section for documentation and installation links for Aparian Slate. The Modbus parameter configuration consists of a general configuration as well as a serial configuration. When downloading this configuration into the module it will be saved in non-volatile memory that persists when the module is powered down.

NOTE: When a firmware upgrade is performed the module will clear all Modbus configuration and routing maps.

The general configuration consists of the following parameters:

Parameter	Description
Instance Name	This parameter is a user defined name to identify between various Modbus Routers.
Description	This parameter is used to provide a more detail description of the application for the module.
Major Revision	The major revision of the module
Mapping Mode	The mapping mode will determine how the Modbus messages are routed. Reactive Tag mode In Reactive Tag mode the module will automatically route the Modbus message and function to the appropriate pre-configured Logix tag. Modbus communication in this mode is initiated by the remote Modbus device. Scheduled Tag mode In Scheduled Tag mode, the Modbus Router will initiate the exchange between the remote Modbus device and Logix. Either by reading data from a Modbus device and writing it into a preconfigured Logix tag or vice versa. Modbus communication in this mode is initiated by the Modbus Router. Unscheduled mode In Unscheduled mode the Modbus Router routes Logix messages to the remote Modbus device and returns the result. There is little configuration required in the Modbus Router as the routing information is configured by Logix for each message transaction. Modbus communication in this mode is initiated by Logix. Refer to the message routing section of the document for a details explanation of the routing operation.

ENIP Retry Limit	The amount of EtherNet/IP retries the module will make once no response was received from the Logix Controller.
ENIP TimeOut	The time in milliseconds after which a retry is sent. Once the first retry is sent the next retry will be sent after the same amount of time. This will repeat until the ENIP Retry Limit is reached.

Table 3.1 - General configuration parameters

The general configuration is shown in the figure below. The Modbus general configuration window is opened by either double clicking on the module in the tree or right-clicking the module and selecting *Configuration*.

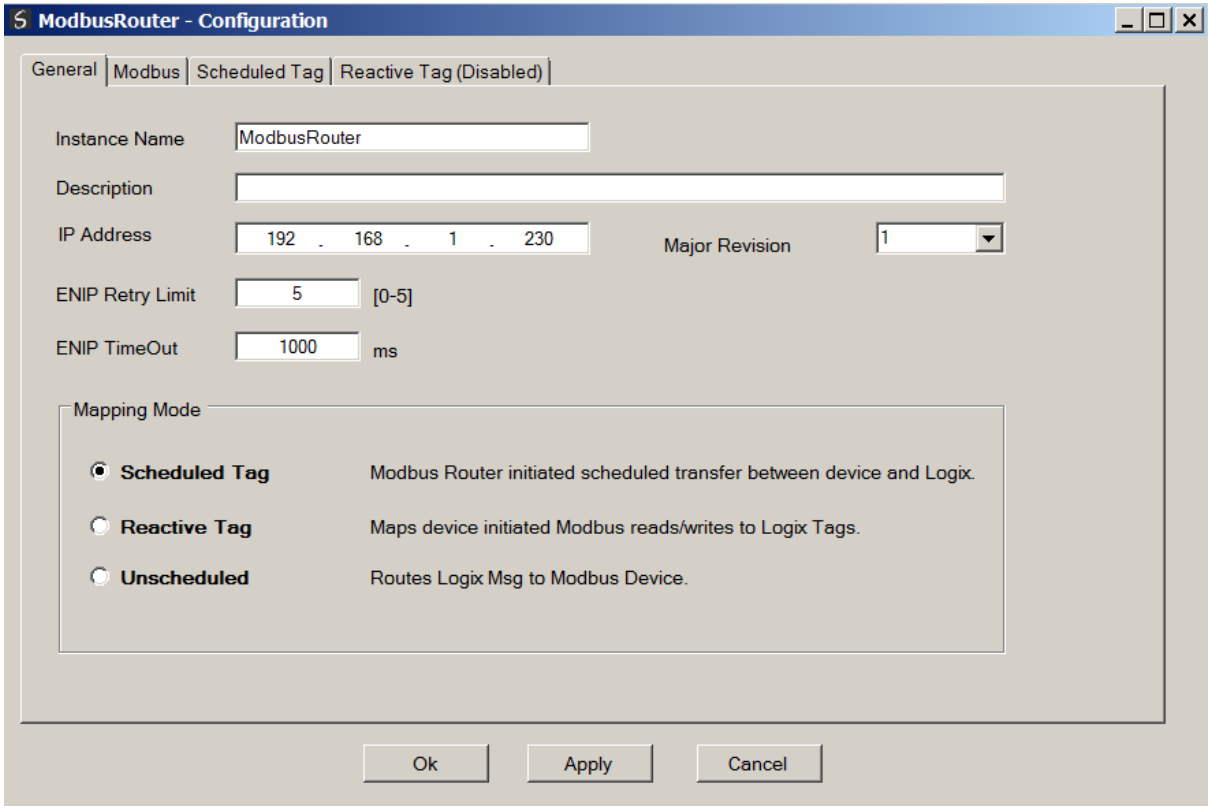

Figure 3.15. - General Configuration

The Modbus configuration consists of the following parameters:

Parameter	**Description**
Protocol	The Modbus Router can interface to the Modbus device(s) on either serial (Modbus-RTU) or Ethernet (Modbus-TCP).
Timeout	The timeout is used to determine the interval between retries when a message exchange has failed.
Reply Wait Time	The reply message wait is the minimum delay before the Modbus reply is transmitted to the Modbus device.
Node Address	The Modbus node address of the Modbus Router.

BAUD Rate	The BAUD rate will configure at what speed the data is send across the RS232 serial network. The module provides the following speeds: 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200
Parity	The parity parameter will configure the parity of the module's RS232 serial port. The module allows for Even, Odd, or None parity setting.

Table 3.2 – Modbus configuration parameters

The Modbus Router can be configured to operate in one of two Base Address modes; Modbus (Base 0) or PLC (Base 1). For example when operating in Base 0 mode the first holding register address will be 40000 whilst in Base 1 mode the first holding register address will be 40001. Depending on the system being interfaced to the user will need to select the correct option. Failing to do this will result in the data being misaligned by one register.

The Modbus configuration is shown in the figure below. The Modbus configuration window is opened by either double clicking on the module in the tree or right-clicking the module and selecting *Configuration.* Once in the configuration window select the second tab at the top *Modbus*.

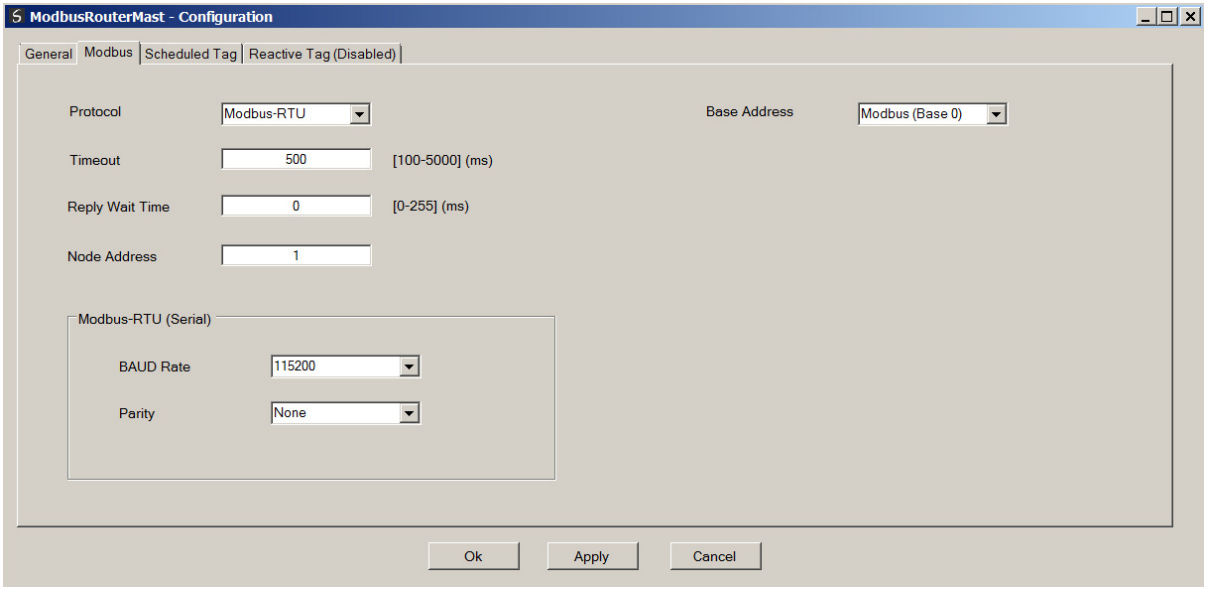

Figure 3.16. – Modbus Configuration

NOTE: If the DF1 Router supports 8 data bits and 1 stop bit.

3.5. MESSAGE ROUTING

The module can be configured to route Modbus data in one of three modes:

- Reactive Tag mode
- Schedule Tag mode
- Unscheduled mode

3.5.1. REACTIVE TAG MODE

The Reactive Tag routing mode allows mapping of virtual Modbus registers to Logix tags across multiple controllers. In this mode the Modbus Router will redirect a Modbus message to a Logix controller at a preconfigured path.

Thus the routing of Modbus read and write register requests is managed by the Modbus Router and converted to direct Logix tag read and write functions.

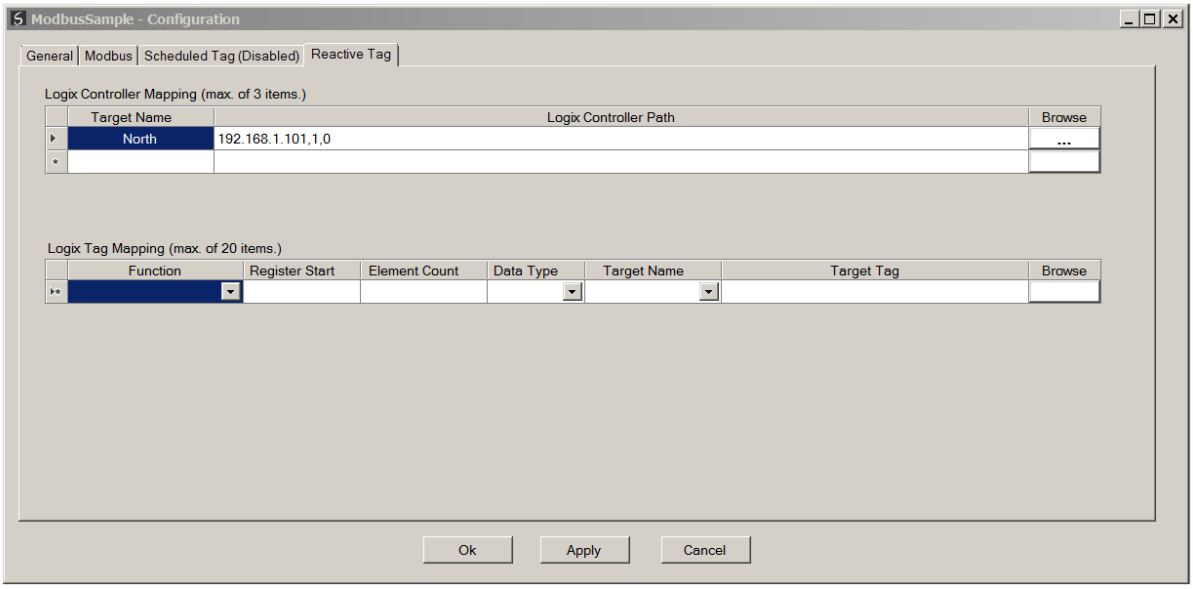

Figure 3.17. - Reactive Tag mode configuration

The Reactive Tag mode is configured in two steps. First the user must create a Target Name (CIP path to the destination Logix controller) which will be used to link the Modbus function and register selection to the destination Logix tag.

The Logix controller paths can either be entered manually or the user can browse to them by clicking the Browse button. The Target Browser will open and automatically scan for all available EtherNet/IP devices.

If the Ethernet/IP module is a bridge module, it can be expanded by right-clicking on the module and selecting the Scan option.

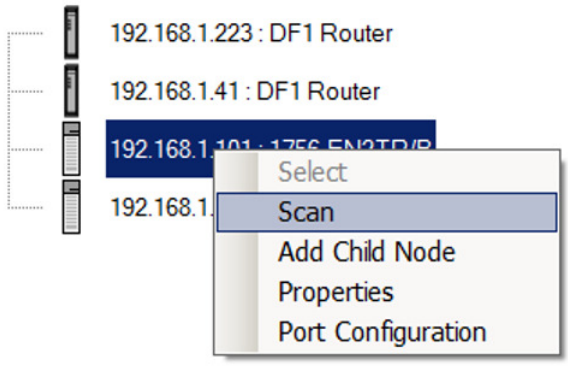

Figure 3.18. - Scanning node in the Target Browser

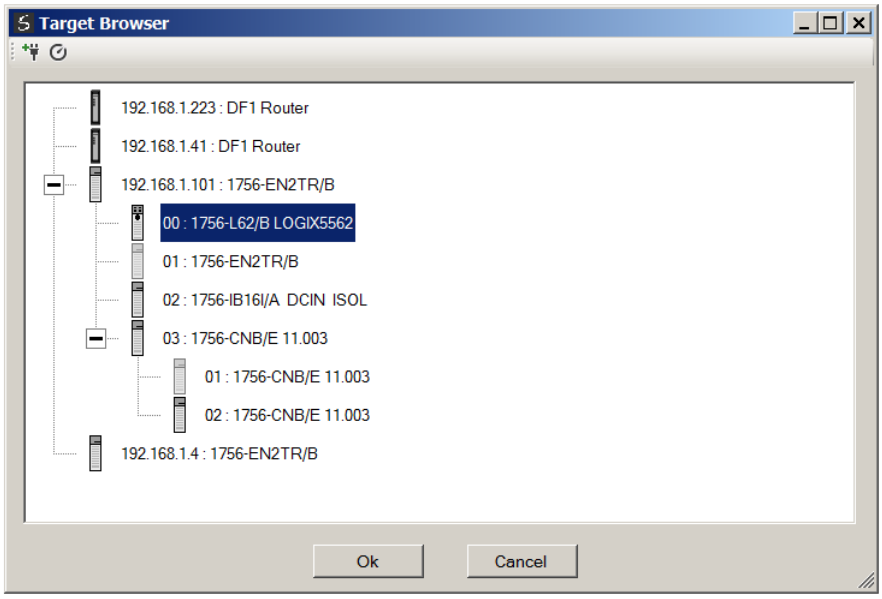

Figure 3.19. - Target Browser selection

The required Logix controller can then be chosen by selecting it and clicking the Ok button, or by double-clicking on the controller module.

A maximum number of 3 controller mapping entries can be added.

The second part of the Reactive Tag mode is to configure the link between a Modbus function and register range to a Logix tag. This will allow the Modbus message initiator to effectively write to, or read from, a Logix tag using traditional Modbus functions.

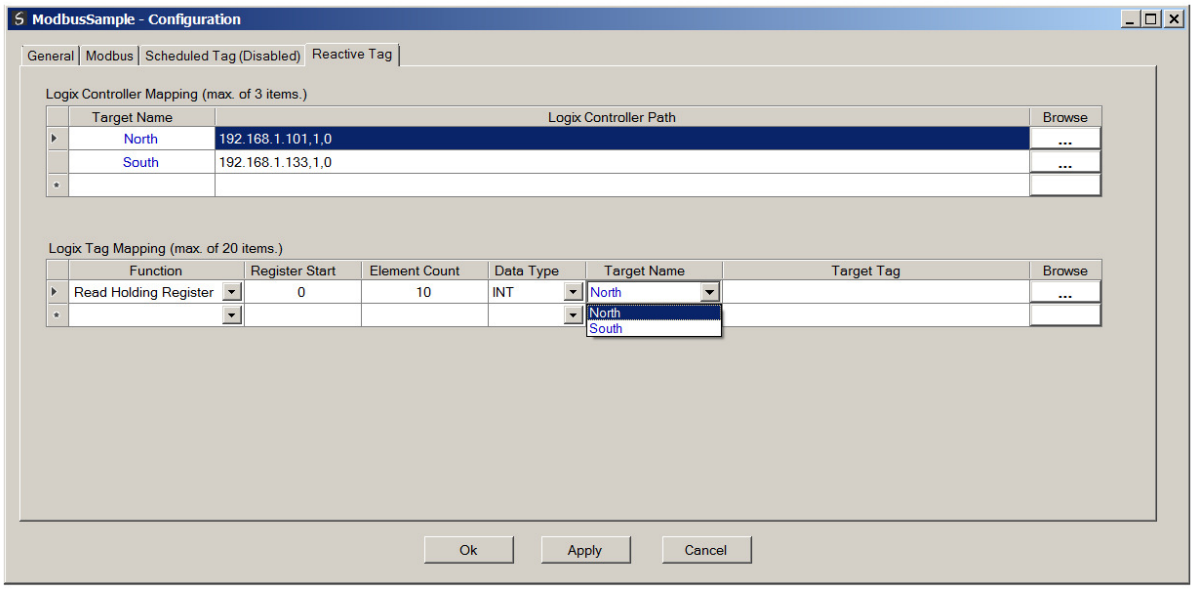

Figure 3.20. – Reactive Tag Mapping

Modbus read and write functions involve the transfer of either Booleans or integers (16 bit). A part or combination of integers may represent other types. For this reason the Modbus Router allows the mapping of integer based functions to the following Logix data types:

- SINT
- INT
- DINT
- REAL

All discrete or Boolean based function need to be mapped to a Logix BOOL array.

NOTE: When mapping a range of registers to a Logix array, it is important to ensure that the destination Logix array is sufficiently large to accommodate all the data.

NOTE: When mapping a range of registers to a Logix DINT or REAL array, the Register Start must be an even number, else the integer data will not be aligned with the 32-bit destination Logix tag.

NOTE: When writing to a DINT or REAL datatype the user cannot write a single Modbus element (16-bit). At least two elements will need to be written to either of the above 32-bit datatypes (DINT or REAL).

NOTE: When doing a single coil write to a Logix Bool array datatype the remaining bits of the 32-bit aligned Bool array will be overwritten.

Next the range of Modbus data to be accessed must be specified. This is achieved by selecting a *Register Start* and *Element Count*.

If the Modbus *Element Count* is 1, it is possible to map a single integer register to a single integer tag. All other combinations will require a Logix tag array to be selected.

Below is an example of the target tag selection. The Target Tag can be either entered manually or selected using the Tag Browser in Slate. The Tag Browser requires the controller to be available on the network.

To browse to the tag, click on the Browse button. The Tag Browser will open and scan all the tags inside that controller. If the controller has been recently scanned in this Slate session, then a cached version of the tags will be displayed. A rescan of the tags can be triggered by selecting the Refresh button in the Tag Browser's toolbar.

Only tags of a relevant type will be enabled, guiding the user to select a suitable tag.

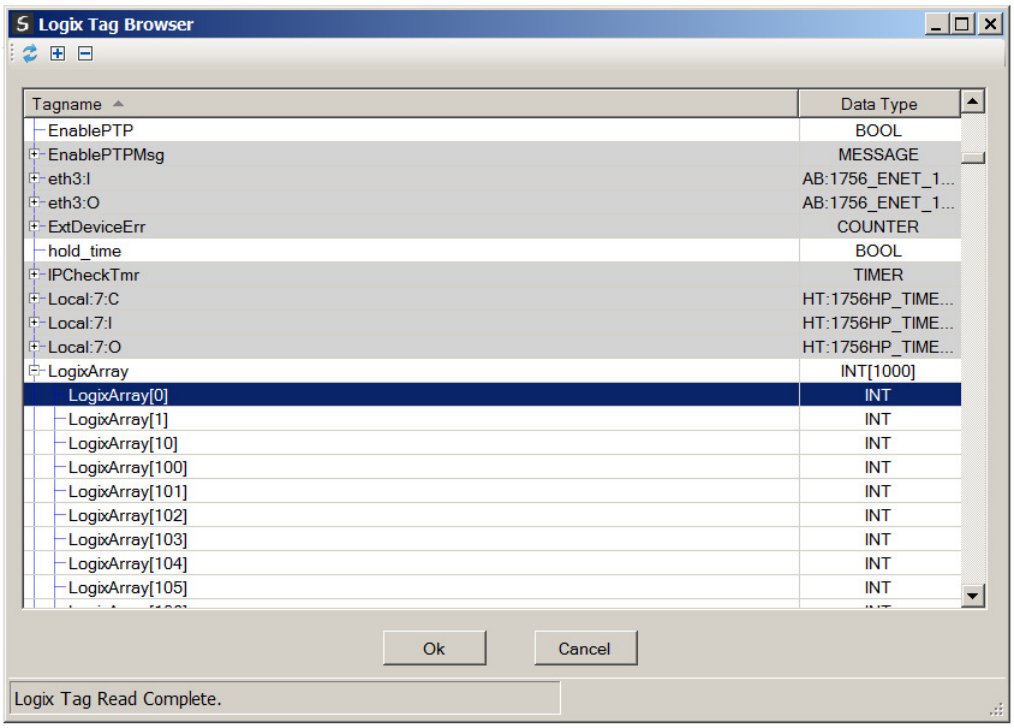

Figure 3.21. – Tag Browser tag selection

The figure below is an example of how Modbus messages are routed to the Logix tags using the Reactive Tag Map mode.

NOTE: It is the user's responsibility to ensure that the Logix tag data type and size matches that of the Modbus message requests. Failing to do this can cause unexpected results and communication faults.

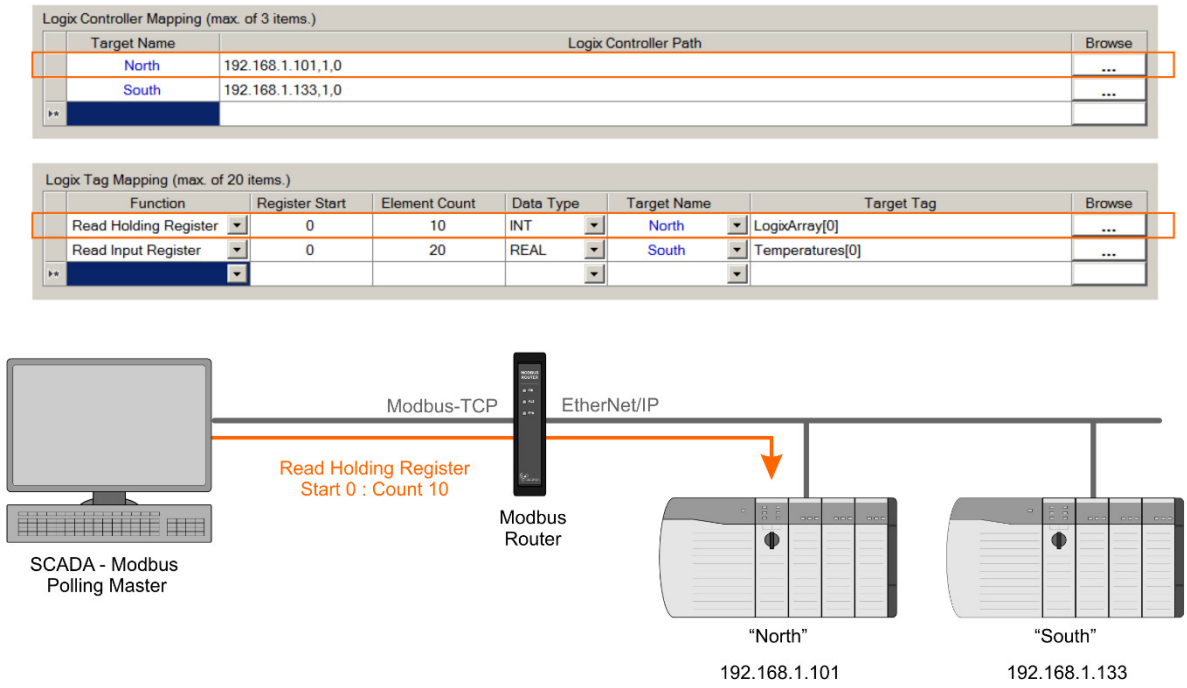

Figure 3.22. - Reactive Tag mode configuration in Slate (example route 1)

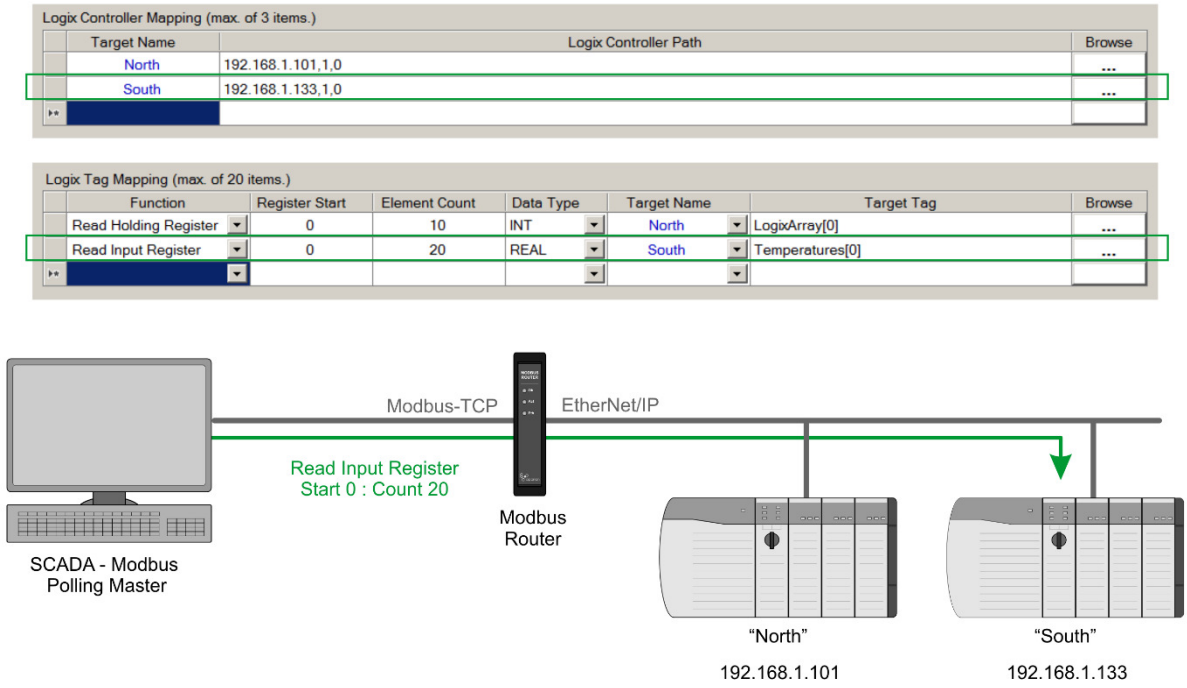

Figure 3.23. - Reactive Tag mode configuration in Slate (example route 2)

3.5.2. SCHEDULED TAG MODE

The Scheduled Tag routing mode transfers data between Modbus devices and Logix controllers. Unlike the Reactive tag mode, the Modbus Router (when in the Scheduled Tag mode) initiates the messaging.

In this mode the Modbus Router transfers data between a Logix controller and Modbus devices without any configuration or programming required in either the Modbus device or the Logix controller. The data will be exchanged at a fixed interval which is configured in Slate.

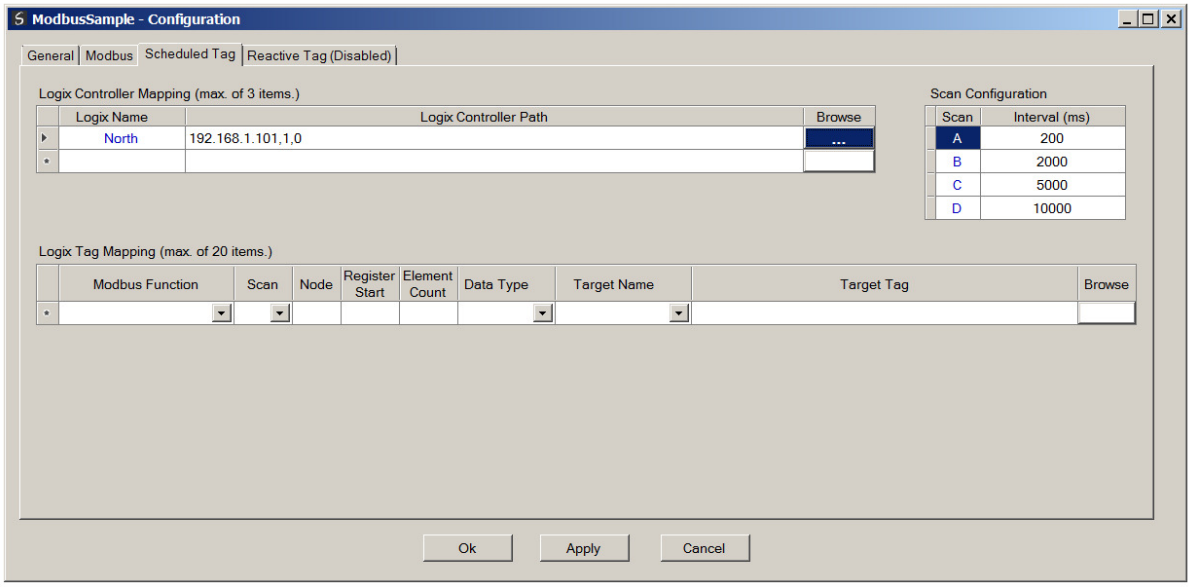

Figure 3.24. - Scheduled Tag configuration

The Schedule Tag mode is configured in three steps. First the user must create a Target Name (CIP path to the destination Logix controller) which will be used to link the Modbus function and register range to the destination Logix tag.

The Logix controller paths can either be entered manually or the user can browse to them by clicking the Browse button. The Target Browser will open and automatically scan for all available EtherNet/IP devices.

If the Ethernet/IP module is a bridge module, it can be expanded by right-clicking on the module and selecting the Scan option.

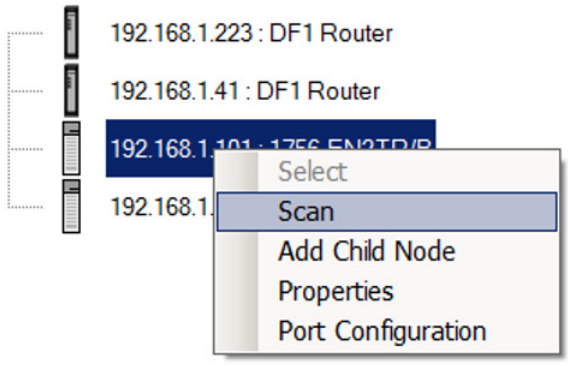

Figure 3.25. - Scanning node in the Target Browser

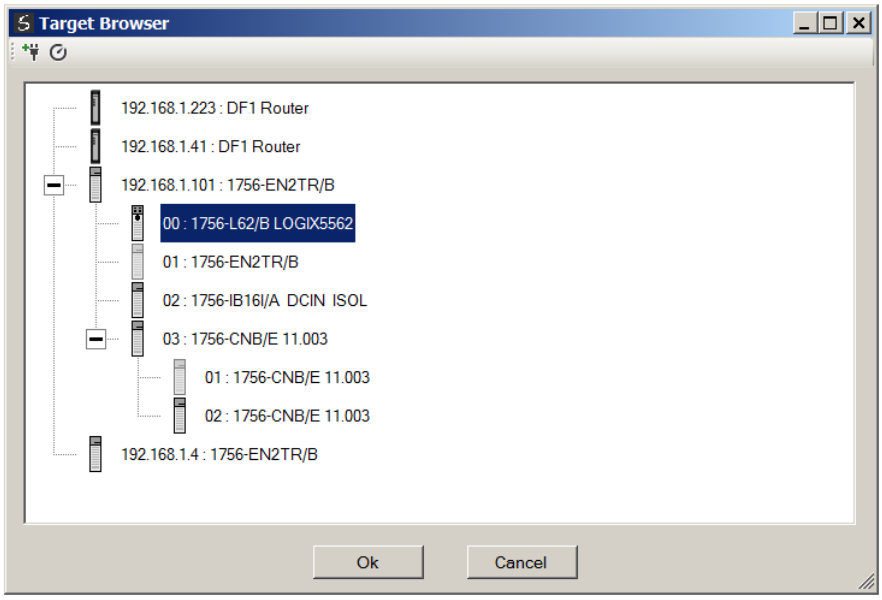

Figure 3.26. - Target Browser selection

The required Logix controller can then be chosen by selecting it and clicking the Ok button, or by double-clicking on the controller module.

A maximum number of 3 controller mapping entries can be added.

The second part of the Scheduled Tag mode setup is to configure the scan intervals. The scan intervals allow different data items to be transferred at different rates. There are 4 scan classes, viz. A, B, C and D. The intervals for each can be adjusted by entering the scan time in milliseconds. The interval must be between 200 milliseconds and 60 seconds.

The third part of the Scheduled Tag mode setup is to configure the link between a Modbus function and register range combination to a Logix tag, and the associated action and scan required.

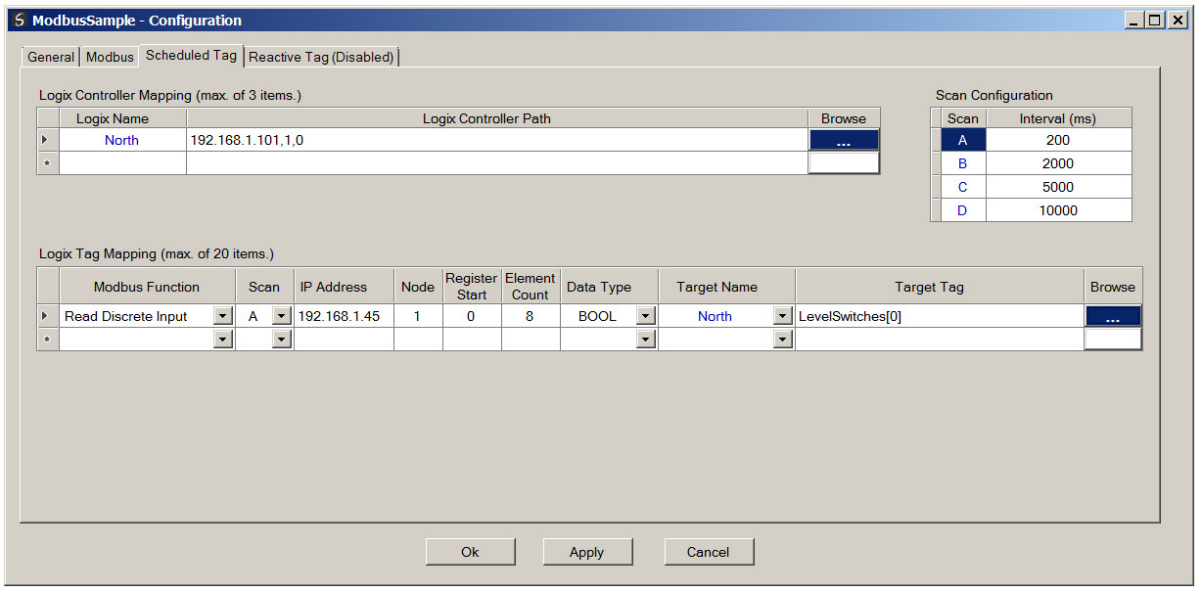

Figure 3.27. – Scheduled Tag Mapping

The Modbus Function is used to determine the actual Modbus command to be sent to the device, as described in the table below:

Selected Function	Modbus Data Type	Single / Multiple	Modbus Function
Read Coil	Boolean	Both	01 - Read Coils
Read Discrete Input	Boolean	Both	02 - Read Discrete Inputs
Read Holding Register	Integer	Both	03 - Read Holding Registers
Read Input Register	Integer	Both	04 - Read Input Registers
Write Coil	Boolean	Single	05 - Write Single Coil
		Multiple	15 - Write Multiple Coils
Write Register	Integer	Single	06 - Write Single Register
		Multiple	16 - Write Multiple Registers

Table 3.3 - Modbus Function Mapping

The Scan field specifies at what rate the transaction will be executed. Select a scan class letter that matches the required interval. Care must be taken to select a realistic scan interval, taking into account the configured Baud rate and message size.

The IP address and Node is the remote device's IP address and node number.

NOTE: The IP address is only relevant (and visible) if the Modbus-TCP protocol is selected. The Node address is relevant for both Modbus protocols.

The range of data to be accessed from the remote device must also be specified. This is achieved by entering the *Register Start* and *Element Count* values.

Modbus read and write functions involve the transfer of either Booleans or integers (16 bit). A part or combination of integers may represent other types. For this reason the Modbus Router allows the mapping of integer based functions to the following Logix data types:

- SINT
- INT
- DINT
- REAL.

All discrete or Boolean based function need to be mapped to a Logix BOOL array.

NOTE: When mapping a range of registers to a Logix array, it is important to ensure that the destination Logix array is sufficiently large to accommodate all the data.

NOTE: When mapping a range of registers to a Logix DINT or REAL array, the Register Start must be an even number, else the integer data will not be aligned with the 32-bit destination Logix tag.

NOTE: When writing to a DINT or REAL datatype the user cannot write a single Modbus element (16-bit). At least two elements will need to be written to either of the above 32-bit datatypes (DINT or REAL).

NOTE: When doing a single coil write to a Logix Bool array datatype the remaining bits of the 32-bit aligned Bool array will be overwritten.

One of the Target Names configured in the first step can be selected by means of the target Name combo box.

The Target Tag can be either entered manually or selected using the Tag Browser in Slate. The Tag Browser requires the controller to be available on the network.

To browse to the tag, click on the Browse button. The Tag Browser will open and scan all the tags inside that controller. If the controller has been recently scanned in this Slate session, then a cached version of the tags will be displayed. A rescan of the tags can be triggered by selecting the Refresh button in the Tag Browser's toolbar.

Only tags of a relevant type will be enabled, guiding the user to select a suitable tag.

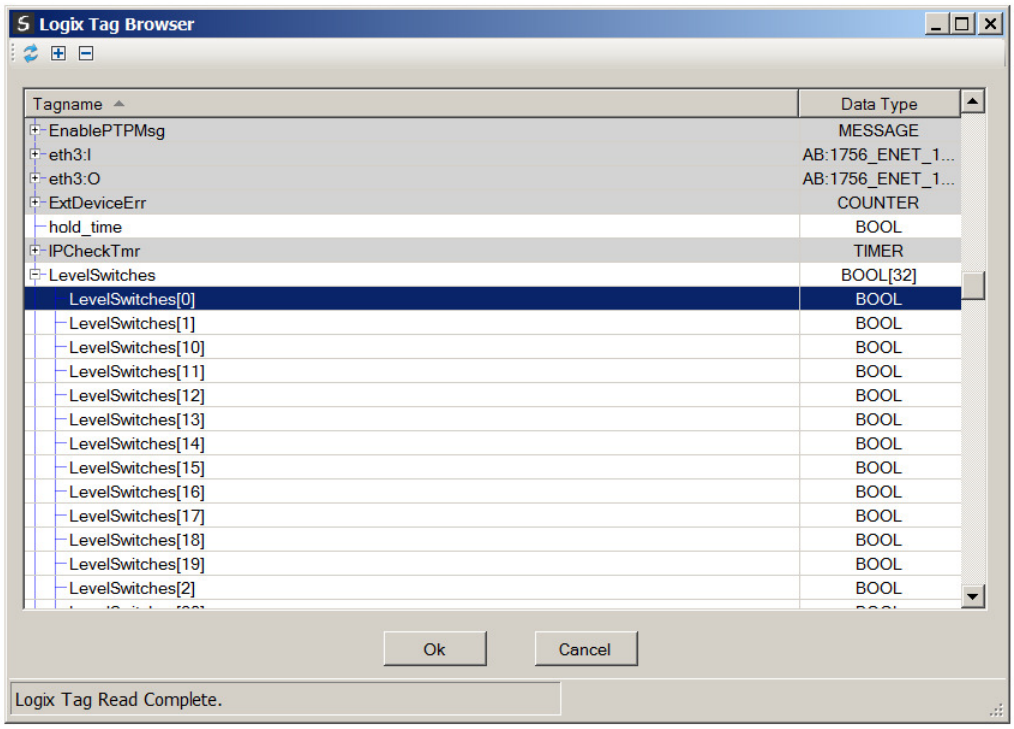

Figure 3.28. – Tag Browser tag selection

3.5.3. UNSCHEDULED MODE

There is no additional configuration required when using the Unscheduled Mode. The configuration required for the Modbus message is contained within the Message Block data, configured in Logix. **Refer** to the operation section for more information.

3.6. MODULE DOWNLOAD

Once the Modbus configuration has been completed, it must be downloaded to the module.

Before downloading the Connection Path of the module should be set. This path will automatically default to the IP address of the module, as set in the module configuration. It can however be modified, if the Modbus Router is not on a local network.

The Connection path can be set by right-clicking on the module and selecting the Connection Path option.

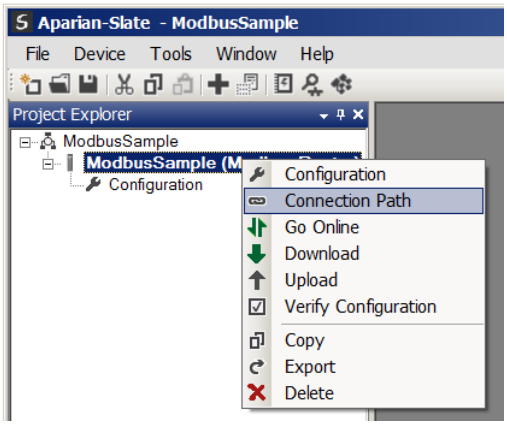

Figure 3.29. - Selecting Connection Path

The new connection path can then be either entered manually or selected by means of the Target Browser.

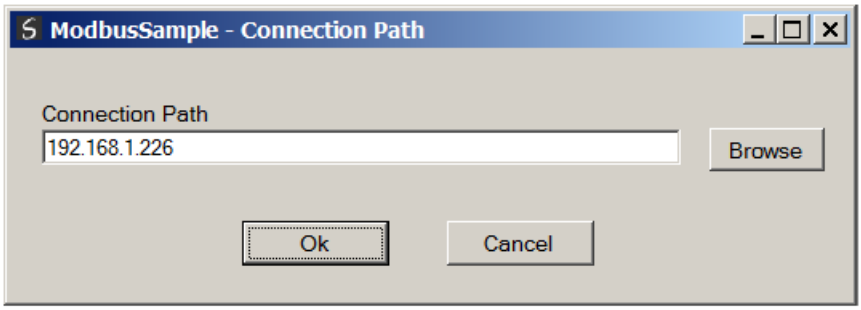

Figure 3.30. - Connection Path

To initiate the download, right-click on the module and select the Download option.

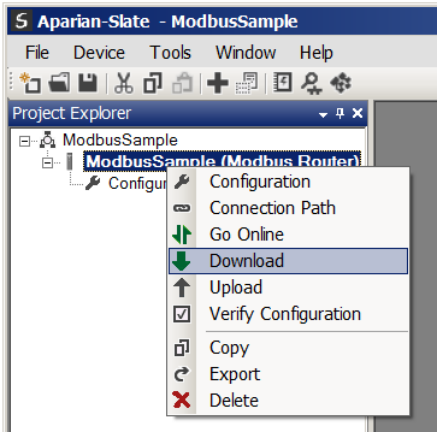

Figure 3.31. - Selecting Download

Once complete, the user will be notified that the download was successful.

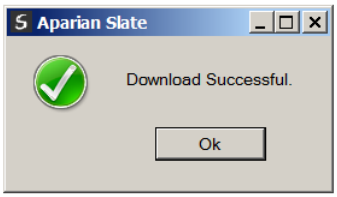

Figure 3.32. - Successful download

During the download process the module's time will be compared to that of the PC's time. Should the difference be greater than 30 seconds, the user will be prompted to set the module time to that of the PC time.

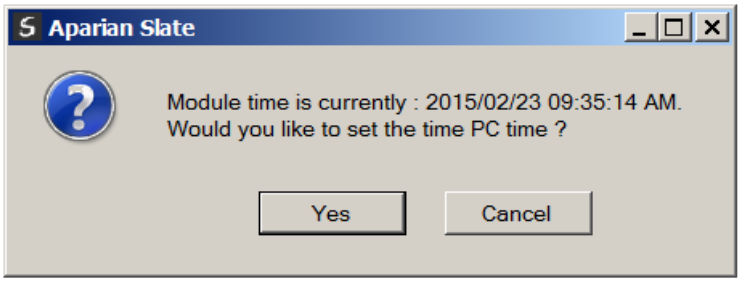

Figure 3.33. – Setting module time

The module time is used only for the event log.

Within the Slate environment the module will be in the Online state, indicated by the green circle around the module.

The module is now configured and will start operating immediately.

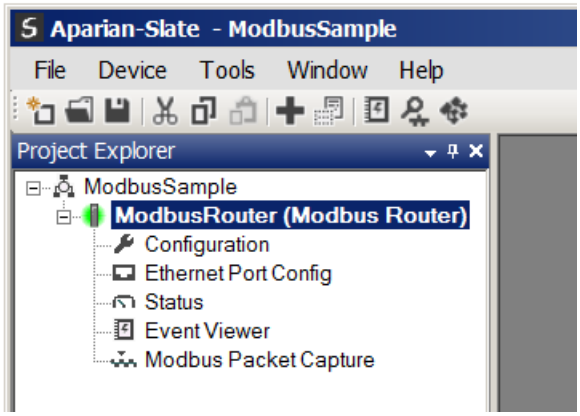

Figure 3.34. - Module online

3.7. RSLOGIX 5000 CONFIGURATION

3.7.1. ADD MODULE TO I/O CONFIGURATION

The module can operate in both a Logix “owned” and standalone mode. When the module operates in a Logix “owned” mode the Modbus Router will need to be added to the RSLogix 5000 IO tree. The module will need to be added as a generic Ethernet module. This is done by right clicking on the Ethernet Bridge in the RSLogix 5000 and selecting *New Module* after which the *ETHERNET-MODULE* is selected to be added as shown in the figure below.

NOTE: See the next section for importing the configuration (L5X).

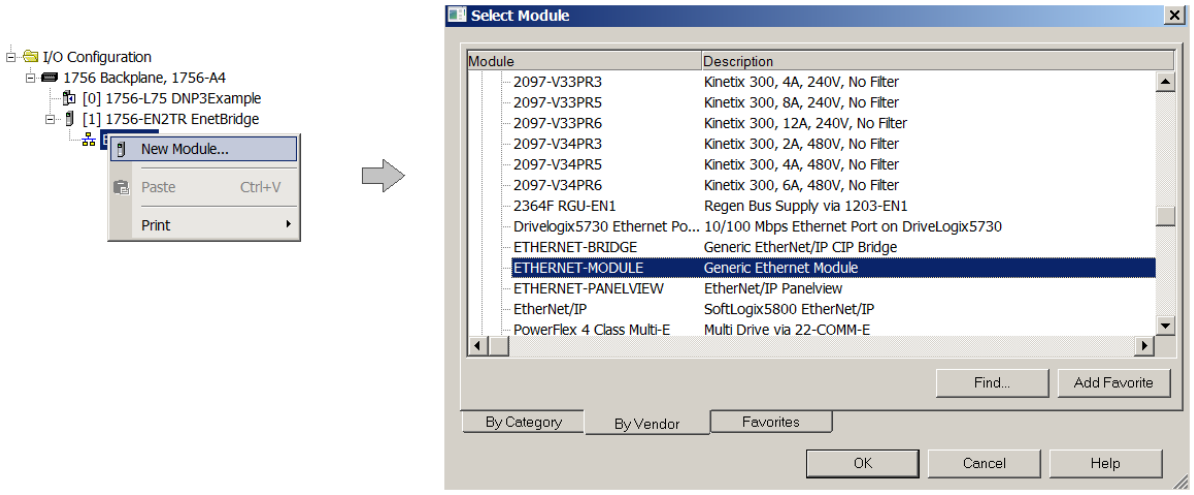

Figure 3.35. - Add a Generic Ethernet Module in RSLogix 5000

The user must enter the IP address of the Modbus Router that will be used. The assembly instance and size must also be added for the input, output, and configuration in the connection parameters section. Below are the required connection parameters.

Connection Parameter	Assembly Instance	Size
Input	111	34 (32-bit)
Output	101	1 (32-bit)
Configuration	102	0 (8-bit)

Table 3.4 - RSLogix class 1 connection parameters for the Modbus Router

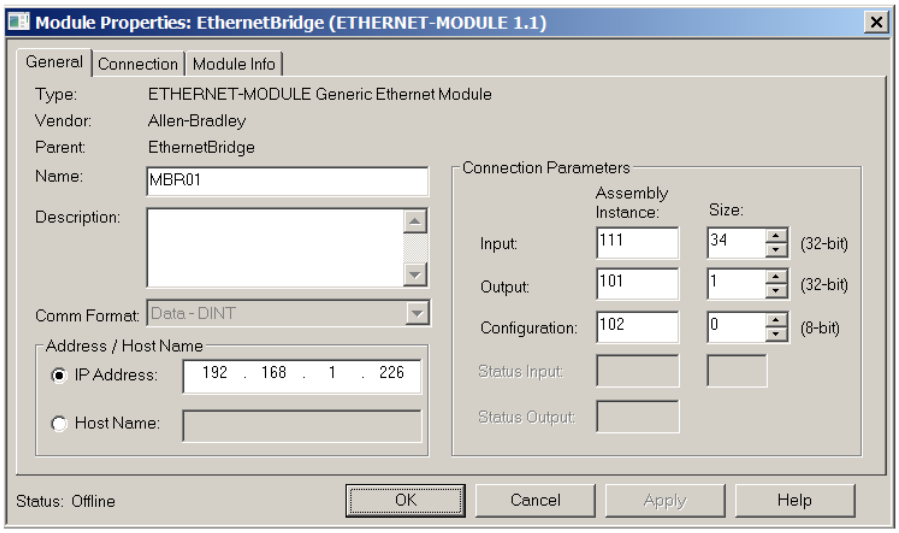

Figure 3.36. - RSLogix General module properties in RSLogix 5000

NOTE: The user will need to enter the exact connection parameters before the module will establish a class 1 connection with the Logix controller.

Next the user needs to add the connection requested packet interval (RPI). This is the rate at which the input and output assemblies are exchanged. The recommended value is 500ms. Refer to the technical specification section in this document for further details on the limits of the RPI.

NOTE: Although the module is capable of running with an RPI of 10ms, it is recommended to set the RPI to 500ms, to avoid unnecessary loading of the module processor.

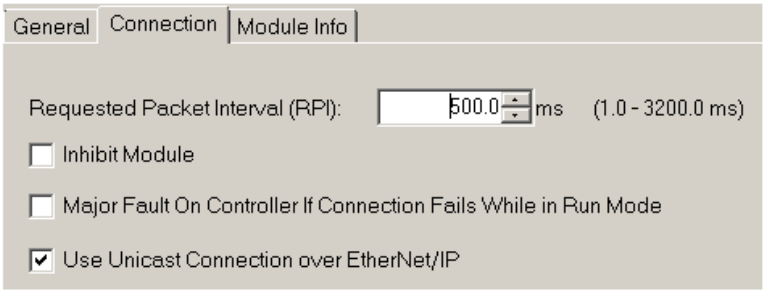

Figure 3.37. - Connection module properties in RSLogix 5000

Once the module has been added to the RSLogix 5000 IO tree the user must assign the User Defined Types (UDTs) to the input and output assemblies. The user can import the required UDTs by right-clicking on *User-Defined* sub-folder in the *Data Types* folder of the IO tree and selecting *Import Data Type*. The assemblies are then assigned to the UDTs with a ladder copy instruction (COP) as shown in the figure below.

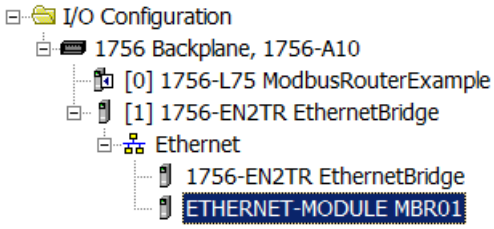

Figure 3.38. – RSLogix 5000 I/O module tree

3.7.2. Importing UDTs and Mapping Routines

To simplify the mapping of the input image, an RSLogix 5000 Routine Partial Import (L5X) file is provided. This file can be imported by right-clicking on the required Program and selecting the Import Routine option.

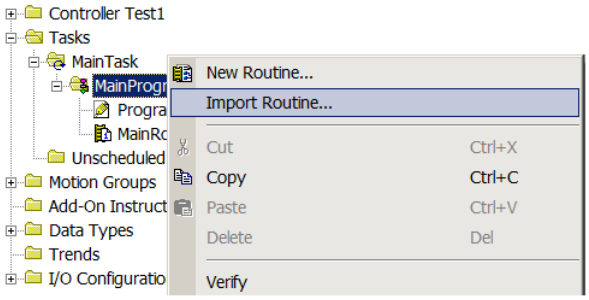

Figure 3.39. – RSLogix 5000 Importing Modbus Router specific routine and UDTs

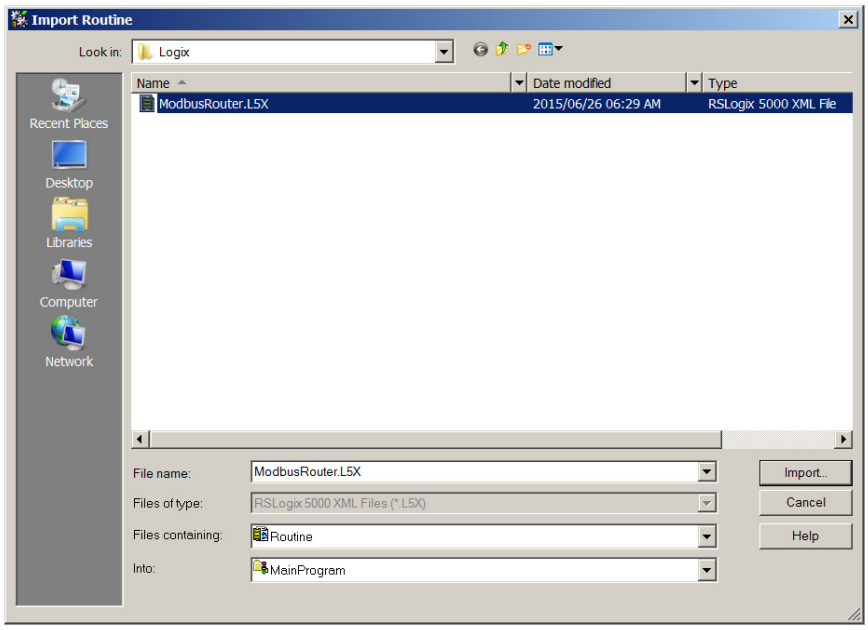

Figure 3.40. - Selecting partial import file

The import will create the following:

- The required UDTs (user defined data types)
- Two controller tags representing the Input and Output assemblies.
- A routine mapping the ModbusRouter module to the aforementioned tags.
- An example Unscheduled Message instruction with the associated Tags

The user may need to change the routine to map to the correct Modbus Router module instance name, and make sure that the mapping routine is called by the Program's Main Routine.

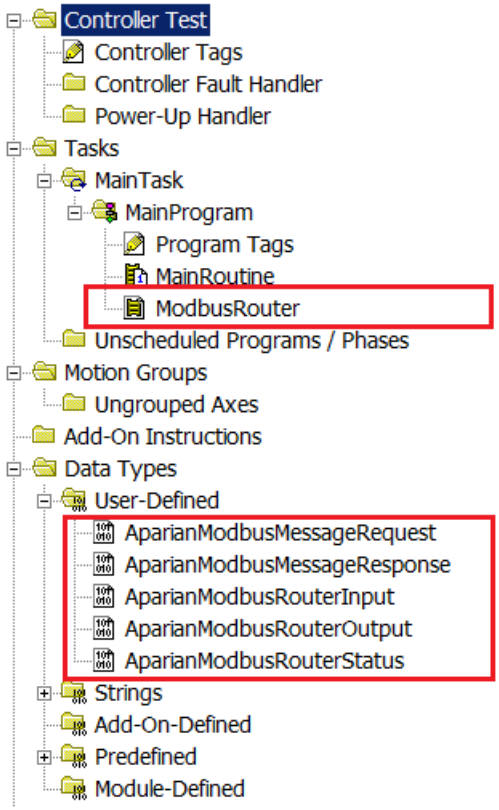

Figure 3.41. - Imported RSLogix 5000 objects

Refer to the additional information section of this document for an example RSLogix 5000 project as well as the required UDTs.

4. OPERATION

4.1. MESSAGE ROUTING

When the module has been correctly setup the Modbus message initiator will send a read/write to a certain Modbus group and variation which will then be routed to a Logix tag. The messages sent by the initiator must be completed with the correct data for successful operation. There are various indicators to determine if the mapping is routing the Modbus messages correctly. Refer to the diagnostics section of this document for a more detailed explanation of the various indicators that can be used to diagnose the module.

4.2. RSLOGIX 5000 ASSEMBLIES

When the module operates in a Logix “owned” mode the Logix controller will establish a class 1 cyclic communication connection with the Modbus Router. A input and output assembly is exchanged at a fix interval. The UDTs provided will convert the input and output arrays into tag based assemblies. Refer to the additional information section in this document for the input and output UDTs.

MBR01Input	{...}	AparianModbusRouterInput
MBR01Input.Instance	'ModbusSample'	STRING
MBR01Input.Status	{...}	AparianModbusRouterStatus
MBR01Input.Status.ReactiveTagMode	0	BOOL
MBR01Input.Status.ScheduledTagMode	0	BOOL
MBR01Input.Status.UnscheduledMode	0	BOOL
MBR01Input.Status.ConfigurationValid	0	BOOL
MBR01Input.Status.RoutingInhibited	0	BOOL
MBR01Input.Status.Reserved1	0	BOOL
MBR01Input.Status.Reserved2	0	BOOL
MBR01Input.Status.Reserved3	0	BOOL
MBR01Input.Status.ScheduledTagStatus0	0	BOOL
MBR01Input.Status.ScheduledTagStatus1	0	BOOL
MBR01Input.Status.ScheduledTagStatus2	0	BOOL
MBR01Input.Status.ScheduledTagStatus3	0	BOOL
MBR01Input.Status.ScheduledTagStatus4	0	BOOL
MBR01Input.Status.ScheduledTagStatus5	0	BOOL
MBR01Input.Status.ScheduledTagStatus6	0	BOOL
MBR01Input.Status.ScheduledTagStatus7	0	BOOL
MBR01Input.Status.ScheduledTagStatus8	0	BOOL
MBR01Input.Status.ScheduledTagStatus9	0	BOOL
MBR01Input.Status.ScheduledTagStatus10	0	BOOL
MBR01Input.Status.ScheduledTagStatus11	0	BOOL
MBR01Input.Status.ScheduledTagStatus12	0	BOOL
MBR01Input.Status.ScheduledTagStatus13	0	BOOL
MBR01Input.Status.ScheduledTagStatus14	0	BOOL
MBR01Input.Status.ScheduledTagStatus15	0	BOOL
MBR01Input.Status.ScheduledTagStatus16	0	BOOL
MBR01Input.Status.ScheduledTagStatus17	0	BOOL
MBR01Input.Status.ScheduledTagStatus18	0	BOOL
MBR01Input.Status.ScheduledTagStatus19	0	BOOL
MBR01Input.TransactionRate	0	DINT
MBR01Input.Temperature	39.5513	REAL
MBR01Input.ModbusRxPacketCount	3	DINT
MBR01Input.ModbusTxPacketCount	5	DINT
MBR01Input.ModbusChecksumErrors	0	DINT
MBR01Input.ModbusTimeout	0	DINT
MBR01Input.ModbusUnspecifiedErrors	0	DINT
MBR01Input.TagReads	0	DINT
MBR01Input.TagWrites	0	DINT
MBR01Input.TagConnectionFailures	0	DINT
MBR01Input.TagErrors	0	DINT

Figure 4.1. - Input assembly UDT structure

1.1.1. Input Assembly

The following parameters are used in the input assembly of the module.

Parameter	Datatype	Description
Instance	STRING	This parameter is the instance name of the module that was configured under the general Modbus configuration in Slate.
Status.ReactiveTagMode	BOOL	Set if the module is operating in Reactive Tag mode.
Status.ScheduledTagMode	BOOL	Set if the module is operating in Scheduled Tag mode.
Status.UnscheduledMode	BOOL	Set if the module is operating in Unscheduled mode.

Status.ConfigurationValid	BOOL	Set if a valid configuration is executing in the module.
Status.RoutingInhibited	BOOL	Set when the module's routing function has been inhibited. Routing can be inhibited by setting a bit in the output assembly of the module.
Status.Reserved1…3	BOOL	Reserved.
Status.ScheduledTagStatus0…19	BOOL[20]	Each bit represents the status of the last scheduled transaction for that specific map item. A true value indicates success.
TransactionRate	DINT	The transaction rate is the number of Modbus messages per second that the module is currently routing.
Temperature	REAL	The internal temperature of the module.
ModbusRxPacketCount	DINT	The total number of Modbus packets received by the module.
ModbusTxPacketCount	DINT	The number of Modbus packets sent by the module.
ModbusChecksumErrors	DINT	The number of corrupted Modbus packets received by the module.
ModbusTimeout	DINT	The number of timed-out Modbus packets sent by the module. Thus no reply was received.
ModbusUnspecifiedErrors	DINT	The number of Modbus errors not defined in any other statistic.
TagReads	DINT	The total number of Logix tag reads executed by the module.
TagWrites	DINT	The total number of Logix tag writes executed by the module.
TagConnectionFailures	DINT	The number of failed class 3 connection attempts. Tag reading and writing requires the module to first establish a class 3 connection with the Logix Controller.
TagErrors	DINT	The number of failed tag access (read/write). These may include privileged violations, non-existing tags, etc.

Table 4.1 - RSLogix 5000 input assembly parameters

1.1.2. Output Assembly

The following parameters are used in the output assembly of the module.

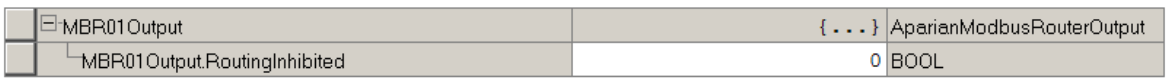

Figure 4.2. - Output assembly UDT structure

Parameter	Datatype	Description
RoutingInhibit	BOOL	This bit inhibits the module routing capabilities. When set, no Modbus messages will be routed. This may be required in applications running a redundant Modbus network where one of the Modbus Routers is to run in a hot-standby mode.

Table 4.2 - RSLogix 5000 output assembly parameters

4.3. UNSCHEDULED MESSAGING

When the Modbus Router is configured in Unscheduled Mode, it will process Modbus message requests sent from Logix via a message instruction.

NOTE: The user will need to build some of the Modbus request data of the unscheduled message.

To simplify the configuration of the required message a number of UDTs have been preconfigured, and are available on the Aparian ModbusRouter webpage (see the further information section).

The message instruction should be setup as follows:

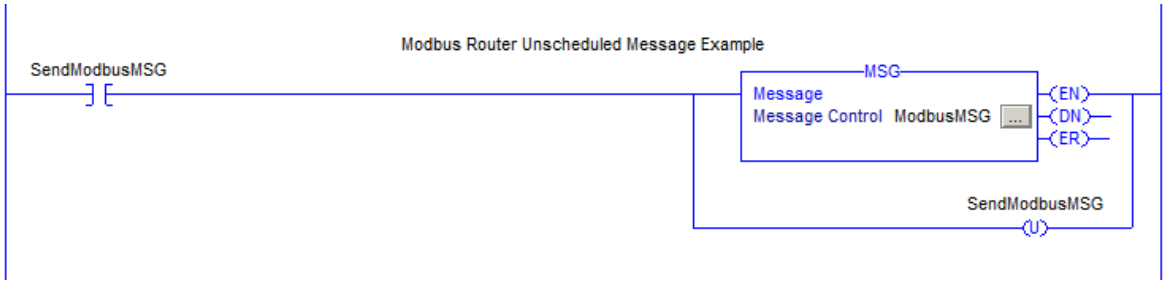

Figure 4.3. - Message Instruction

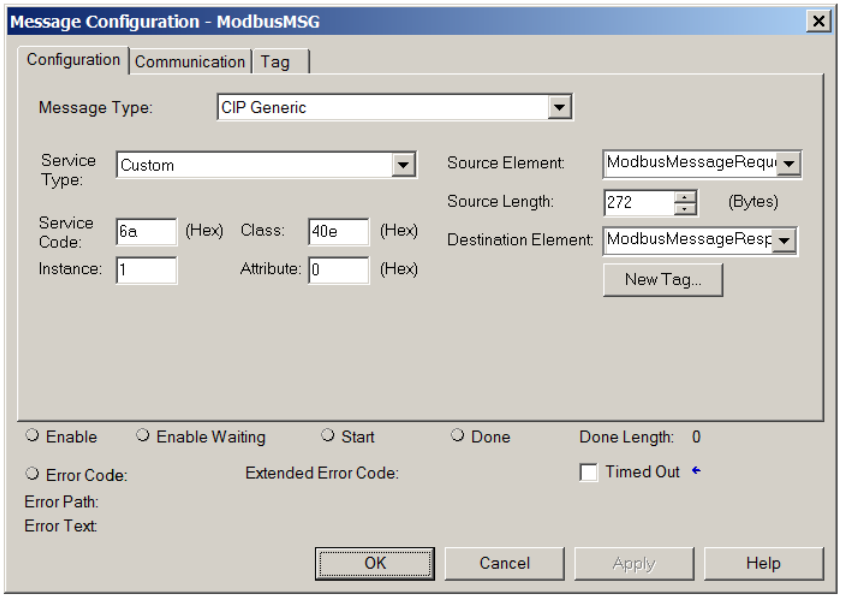

Figure 4.4. - Message Configuration

Parameter	Description
Message Type	CIP Generic
Service Type	Custom
Service Code	6A (Hex) - Unscheduled Modbus Pass-through
Class	40E (Hex)
Instance	1
Attribute	0
Source Element	The request tag instance. Must follow the structure of the AparianModbusMessageRequest UDT.
Source Length	272
Destination Element	The response tag instance. Must follow the structure of the AparianModbusMessageResponse UDT.

Table 4.3. - Message Configuration Paramaters

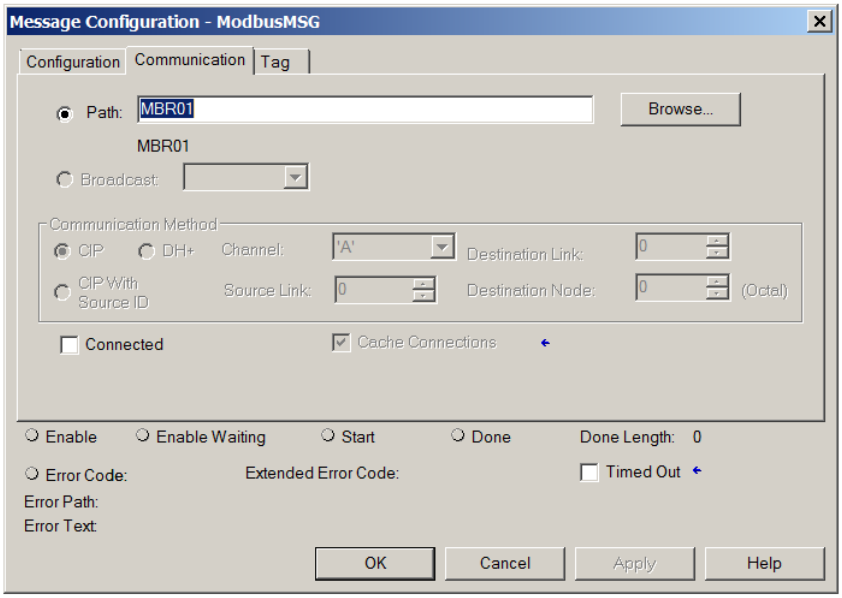

Figure 4.5. - Messsage Configuration - Communication

The Path must be configured to that of the Modbus Router. If the Modbus Router has been added in the I/O tree, then the Browse option can be used to select the path.
Alternatively, enter the CIP path in the format :

1,X,2,IP , where
1 represents the backplane port,
X represents the slot of the Ethernet bridge module,
2 represents the Ethernet port of the Ethernet bridge module and
IP represents the IP address of the Modbus Router.

e.g. **1,1,2,192.168.1.41**

The request tag (e.g. ModbusMessageRequest) should be configures as follows:

ModbusMessageRequest	{ . . . }	AparianModbusMessageRequest
ModbusMessageRequest.DestinationAddress	1	INT
ModbusMessageRequest.Function	3	INT
ModbusMessageRequest.Port	0	INT
ModbusMessageRequest.RequestSize	4	INT
ModbusMessageRequest.IPAddress	{ . . . }	INT[4]
ModbusMessageRequest.IPAddress[0]	192	INT
ModbusMessageRequest.IPAddress[1]	168	INT
ModbusMessageRequest.IPAddress[2]	1	INT
ModbusMessageRequest.IPAddress[3]	219	INT
ModbusMessageRequest.RequestData	{ . . . }	SINT[256]

Figure 4.6. - Unscheduled Message Request Tag

Parameter	Description
Destination Address	The Modbus node address of the destination device.
Function	The Modbus function code. (See the table below for common function codes.)
Port	The Modbus Router supports two ports to interface to a Modbus device: 0 – Serial Port (Modbus-RTU). 1 - Ethernet Port (Modbus-TCP).
Request Data Size	The size of the request data being sent
IP Address	The IP address of the remote device when an Ethernet port was selected above.
Request Data	Modbus request data. All the bytes succeeding the function code but preceding the checksum. See the example further on in the chapter.

Table 4.4. - Unscheduled Message Request Parameters

ModbusMessageResponse	{ . . . }	AparianModbusMessageResponse
ModbusMessageResponse.Status	0	INT
ModbusMessageResponse.ResponseLength	5	INT
ModbusMessageResponse.Response	{ . . . }	SINT[255]

Figure 4.7. - Unscheduled Message Response Tag

Parameter	Description
Status	The Modbus exception code returned. A zero represents success. (See the table below for common exception codes.)
Response Length	Length of the response data received.
Response Data	Response to the Modbus application layer object request.

Table 4.5. - Unscheduled Message Response Parameters

After the message has been executed successfully (Msg.DN) the Response Data will hold the Modbus response data.

Below is a table showing common Modbus Functions:

Code	Modbus Function
1	Read Coils
2	Read Discrete Inputs
3	Read Holding Registers
4	Read Input Registers
5	Write Single Coil
6	Write Single Register
7	Read Exception Status
8	Diagnostic
11	Get Comm Event Counter
12	Get Comm Event Log
15	Write Multiple Coils
16	Write Multiple Registers
17	Report Server ID
20	Read File Record
21	Write File Record
22	Mask Write Register
23	Read/Write Multiple Registers
24	Read FIFO Queue

Table 4.6 - Common Modbus Functions

The following table shows common Status / Exception codes.

Code	Modbus Exception
0	Success
1	Illegal Function
2	Illegal Data Address
3	Illegal Data Value
4	Server Device Failure
5	Acknowledge
6	Server Device Busy
8	Memory Parity Error
10	Gateway Path Unavailable
11	Gateway Target Device Failed to Respond

Table 4.7 - Common Exception Codes

4.3.1. EXAMPLE

In the following example, the unscheduled message instruction is used to read two Holding Registers (0 and 1), from a remote Ethernet Modbus device located at IP address 192.168.1.219.

The standard ModbusMessageRequest structure is populated as shown in the figure below.

⊟ ModbusMessageRequest	{...}	
⊞ ModbusMessageRequest.DestinationAddress	1	Modbus node address of destination device
⊞ ModbusMessageRequest.Function	3	Function : 03 - Read Holding Register
⊞ ModbusMessageRequest.Port	1	Port : 1 - Ethernet (Modbus-TCP)
⊞ ModbusMessageRequest.RequestSize	4	Number of bytes in Request Data
⊟ ModbusMessageRequest.IPAddress	{...}	
⊞ ModbusMessageRequest.IPAddress[0]	192	Modbus IP address of destination device - 192.168.1.219
⊞ ModbusMessageRequest.IPAddress[1]	168	
⊞ ModbusMessageRequest.IPAddress[2]	1	
⊞ ModbusMessageRequest.IPAddress[3]	219	
⊟ ModbusMessageRequest.RequestData	{...}	
⊞ ModbusMessageRequest.RequestData[0]	0	Register Start - 0000
⊞ ModbusMessageRequest.RequestData[1]	0	
⊞ ModbusMessageRequest.RequestData[2]	0	Register Count - 0002
⊞ ModbusMessageRequest.RequestData[3]	2	

Figure 4.8. – Modbus Message Request Example

5. DIAGNOSTICS

5.1. LEDS

The module provides three LEDs for diagnostics purposes as shown in the front view figure below. A description of each LED is given in the table below.

Figure 5.1 - Modbus Router front view

LED	Description
Ok	The module's Ok LED will provide information regarding the system-level operation of the module. Thus if the LED is red then the module is not operating correctly. For example if the module application firmware has been corrupted or there is a hardware fault the module will have a red Module LED. If the LED is green then the module has booted and is running correctly.
Act	The activity LED is used for the RS232 or RS485 serial port. Every time there is a successful Modbus routing transaction the LED will flash green. The LED will flash red if the routing was unsuccessful (e.g. failed checksum).
Eth	The Ethernet LED will light up when an Ethernet link has been detected (by plugging in a connected Ethernet cable). The LED will flash every time traffic was detected.

Table 5.1 - Module LED operation

5.2. MODULE STATUS MONITORING IN SLATE

The Modbus Router can provide a range of statistics which can assist with module operation, maintenance, and fault finding. The statistics can be accessed in full by Slate or using the web server in the module.

To view the module's status in the Aparian-Slate environment, the module must be online. If the module is not already Online (following a recent configuration download), then right-click on the module and select the *Go Online* option.

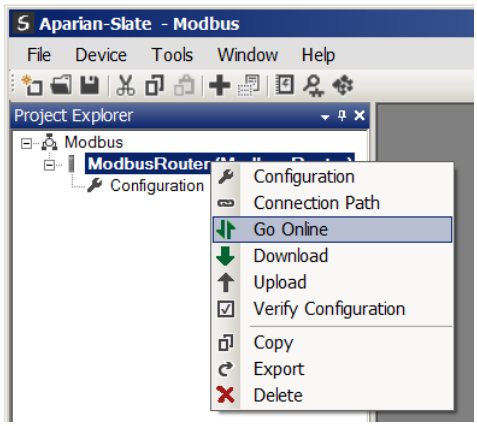

Figure 5.2. - Selecting to Go Online

The Online mode is indicated by the green circle behind the module in the Project Explorer tree.

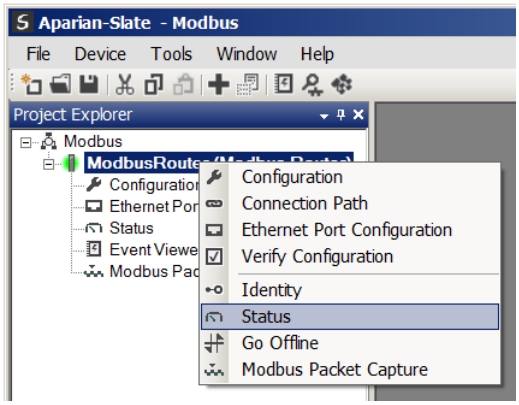

Figure 5.3. - Selecting online Status

The Status monitoring window can be opened by either double-clicking on the *Status* item in the Project Explorer tree, or by right-clicking on the module and selecting *Status*.

The status window contains multiple tabs to display the current status of the module.

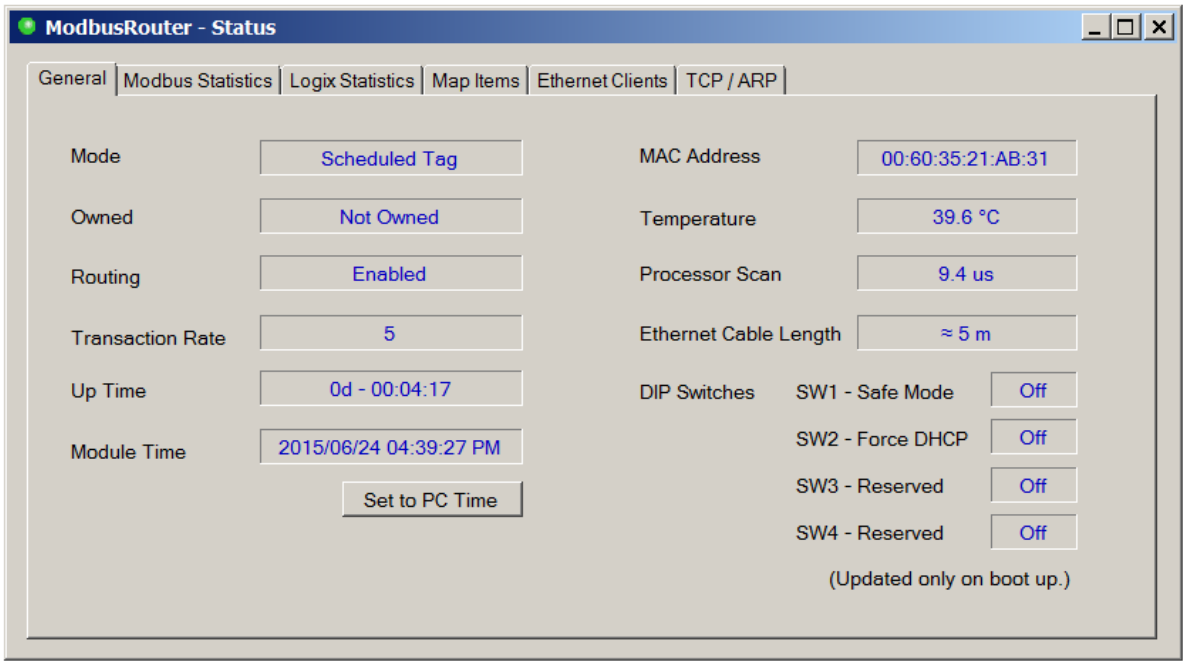

Figure 5.4. - Status monitoring - General

The General tab displays the following general parameters and can also be used to set the module time to the PC time:

Parameter	Description
Mode	Indicates the current operating mode : Reactive Tag, or Scheduled Tag, or Unscheduled.
Owned	Indicates whether or not the module is currently owned (Class 1) by a Logix controller.
Routing	Indicates whether the routing of module is enabled or inhibited. The routing operation can be inhibited in the output assembly of the module.
Transaction Rate	The transaction rate is the number of Modbus messages per second that the module is currently routing.
Up Time	Indicates the elapsed time since the module was powered-up.
Module Time	Indicates the module's internal time. The module time is stored in UTC (Universal Coordinate Time) but displayed on this page according to the local PC Time Zone settings.
MAC Address	Displays the module's unique Ethernet MAC address.
Temperature	The internal temperature of the module.

Processor Scan	The amount of time (microseconds) taken by the module's processor in the last scan.
Ethernet Cable Length	Indicates the estimated length of the Ethernet cable attached to the module. (Accuracy of 5m)
DIP Switch Position	The status of the DIP switches when the module booted. Note that this status will not change if the DIP switches are altered when the module is running.

Table 5.2 - Parameters displayed in the Status Monitoring – General Tab

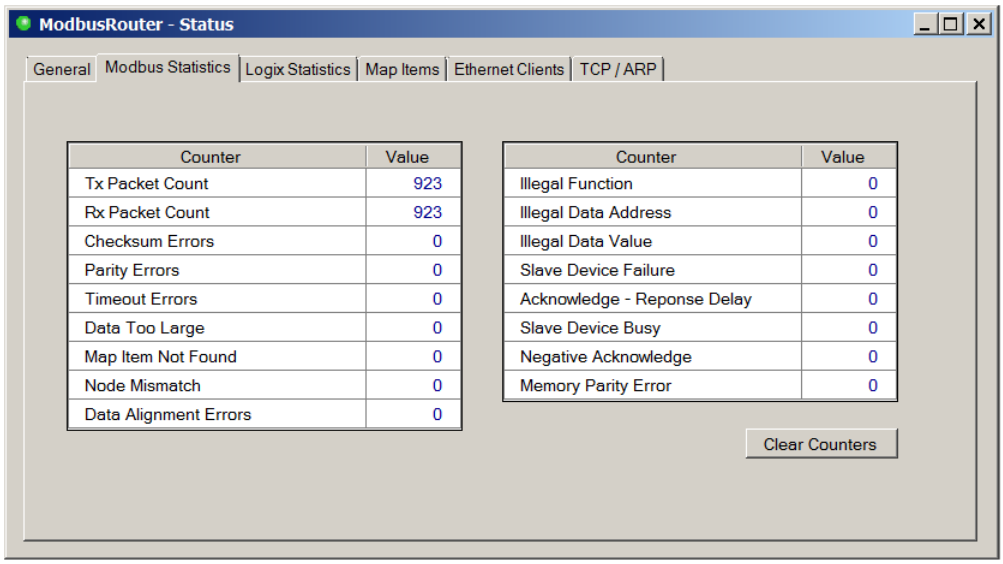

Figure 5.5. - Status monitoring – Modbus Statistics

The Modbus Statistics tab displays the statistics associated with the Modbus communication and mapping.

Statistic	**Description**
Tx Packet Count	The number of Modbus packets sent by the module.
Rx Packet Count	The number of Modbus packets received by the module.
Checksum errors	The number of corrupted Modbus packets received by the module.
Parity errors	The number of bytes with parity errors received by the module.
Timeout Errors	The number of message response timeouts the module has encountered.
Data Too Large	The number of Modbus requests or responses where the data was too large to process.
Map Item Not Found	The number of Modbus requests did not match any mapped items.
Node Mismatch	The received Modbus request did not match the module's Modbus node address.

Data Alignment Errors	The Modbus request and associated mapped item is not byte aligned with the destination Logix tag.
Illegal Function	The number of times the Modbus device responded with an Illegal Function exception.
Illegal Data Address	The number of times the Modbus device responded with an Illegal Data Address exception.
Illegal Data Value	The number of times the Modbus device responded with an Illegal Data Value exception.
Slave Device Failure	The number of times the Modbus device responded with a Device Failure exception.
Acknowledge –Response Delay	The number of times the Modbus device responded with an Acknowledge exception.
Slave Device Busy	The number of times the Modbus device responded with a Slave Busy exception.
Negative Acknowledge	The number of times the Modbus device responded with a Negative Acknowledge exception.
Memory Parity Error	The number of times the Modbus device responded with a Memory Parity exception.

Table 5.3 – Modbus statistics

The following Logix statistics are only relevant when the module is running in either Reactive Tag or Scheduled Tag mode.

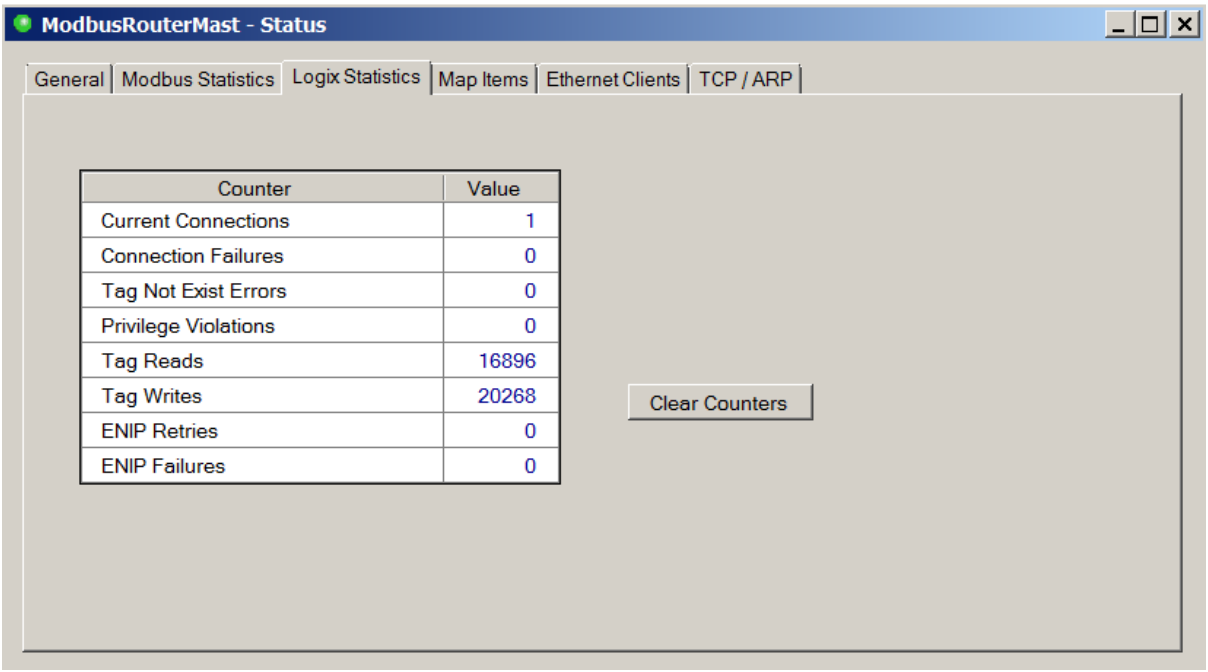

Figure 5.6. - Status monitoring – Logix Statistics

Statistic	Description
Current Connections	The number of current open class 3 connections.
Connection Failures	The number of failed attempts at establishing a class 3 connections with a Logix controller.
Tag Not Exist Errors	The number of tag read and tag write transactions that failed due to the destination tag not existing.
Privilege Violation Errors	The number of tag read and tag write transactions that failed due to a privilege violation error. This may be caused by the External Access property of the Logix tag being set to either None or Read Only.
Tag Reads	The number of tag read transactions executed by the Modbus Router module.
Tag Writes	The number of tag write transactions executed by the Modbus Router module.
ENIP Retries	This count increases when no response was received from the Logix Controller by the time the ENIP timeout is reached.
ENIP Failures	This count increases when the ENIP Retry Limit is reached and no response has been received from the Logix Controller.

Table 5.4 - Tag Mapping statistics

The Map Items tab will display the successful packet counts processed by each mapping item. If an item count changes, then the success count field will be displayed with a green background for approximately 3 seconds. This provides quick visual feedback as to which items are currently active.

The fields in the map items will adjust to suite the appropriate mode.

No items are displayed in Unscheduled mode.

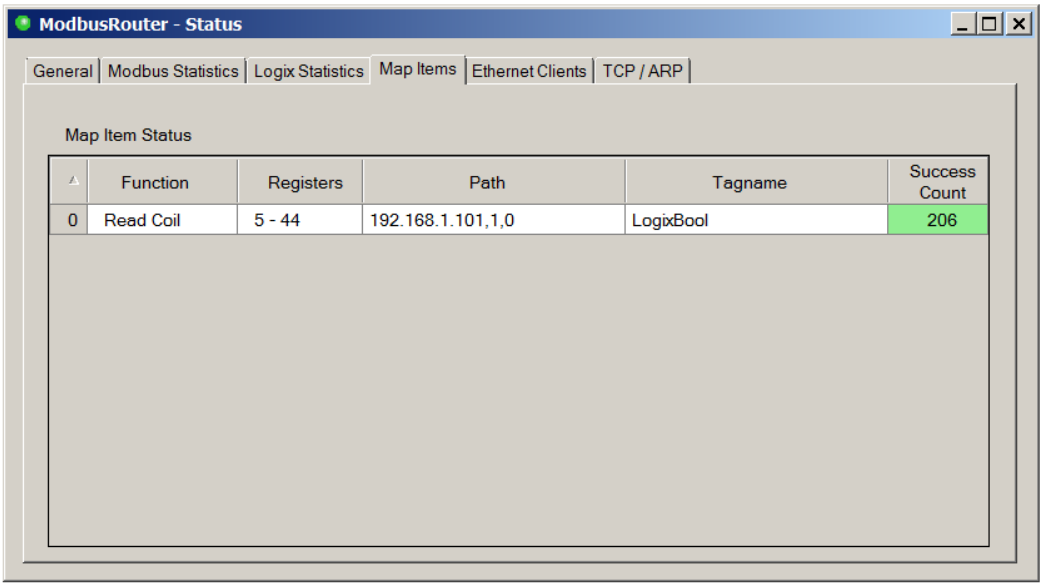

Figure 5.7. - Map Item status

5.3. MODBUS PACKET CAPTURE

The module provides the capability to capture the Modbus traffic for analysis. The will allow the user and the support team to resolve any possible issue on site.

To invoke the capture of the module double-click on the Modbus Packet Capture item in the Project Explorer tree.

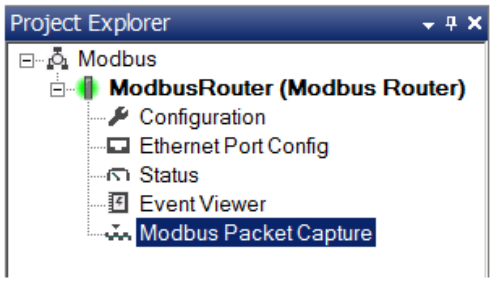

Figure 5.8. - Selecting Modbus Packet Capture

The Modbus Packet Capture window will open and automatically start capturing all Modbus packets.

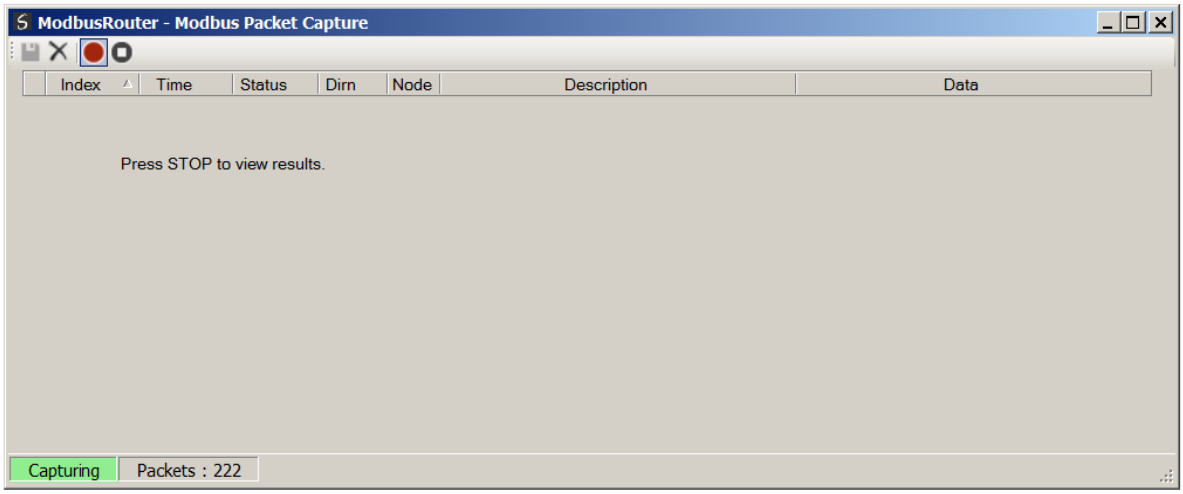

Figure 5.9. – Modbus packet capture

To display the captured Modbus packets, the capture process must first be stopped, by pressing the Stop button.

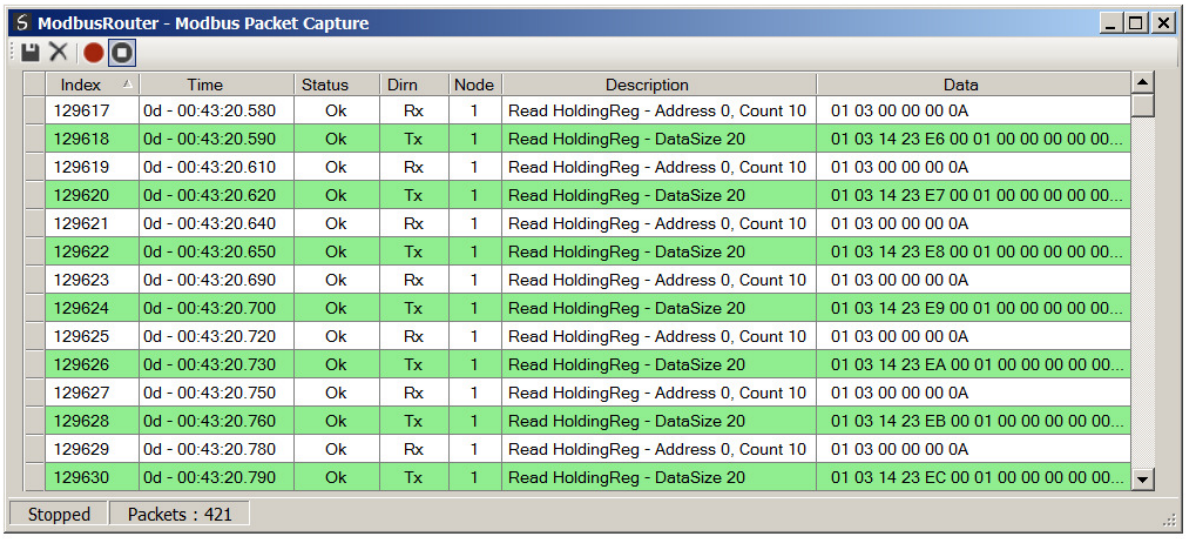

Figure 5.10. – Modbus Packet Capture complete

The captured Modbus packets are tabulated as follows:

Statistic	Description
Index	The packet index, incremented for each packet sent or received.
Time	The elapsed time since the module powered up.
Status	The status of the packet. Received packets are checked for valid Modbus constructs and valid checksums.
Dirn	The direction of the packet, either transmitted (Tx) or received (Rx).
Node	Modbus node address of the message destination.
Description	A brief description of the packet, showing the function and register range if applicable.
Data	The raw packet data.

Table 5.5. – Modbus Packet Capture fields

The packet capture can be saved to a file for further analysis, by selecting the Save button on the toolbar. Previously saved Modbus Packet Capture files can be viewed by selecting the Modbus Packet Capture Viewer option in the tools menu.

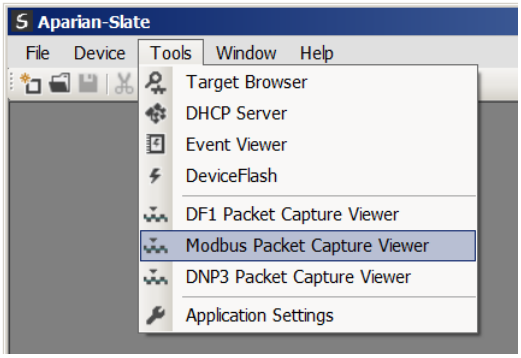

Figure 5.11. - Selecting the Modbus Packet Capture Viewer

5.4. MODULE EVENT LOG

The Modbus Router module logs various diagnostic records to an internal event log. These logs are stored in non-volatile memory and can be displayed using Slate or via the web interface.
To vie them in Slate, select the Event Viewer option in the Project Explorer tree.

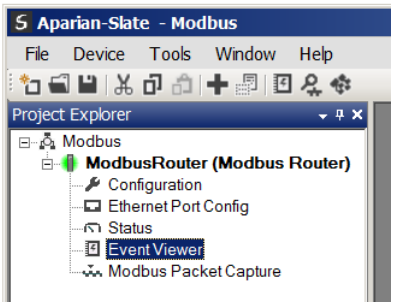

Figure 5.12. - Selecting the module Event Log

The Event Log window will open and automatically read all the events from the module.
The log entries are sorted so as to have the latest record at the top. Custom sorting is achieved by double-clicking on the column headings.

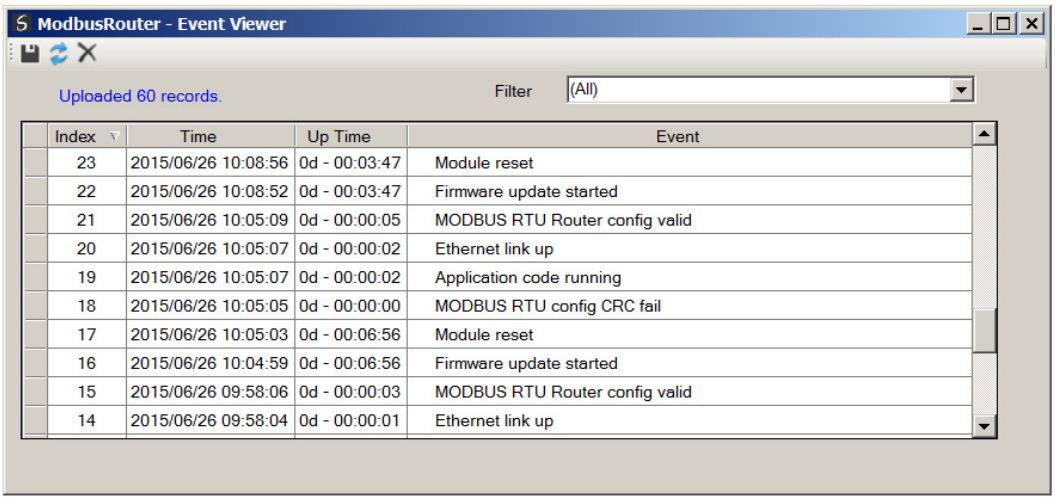

Figure 5.13. – Module Event Log

The log can also be stored to a file for future analysis, by selecting the Save button in the tool menu.

To view previously saved files, use the Event Log Viewer option under the tools menu.

5.5. WEB SERVER

The Modbus Router provides a web server allowing a user without Slate or RSLogix 5000 to view various diagnostics of the module. This includes Ethernet parameters, system event log, advanced diagnostics, and application diagnostics (Modbus diagnostics).

NOTE: The web server is view **only** and thus no parameters or configuration can be altered from the web interface.

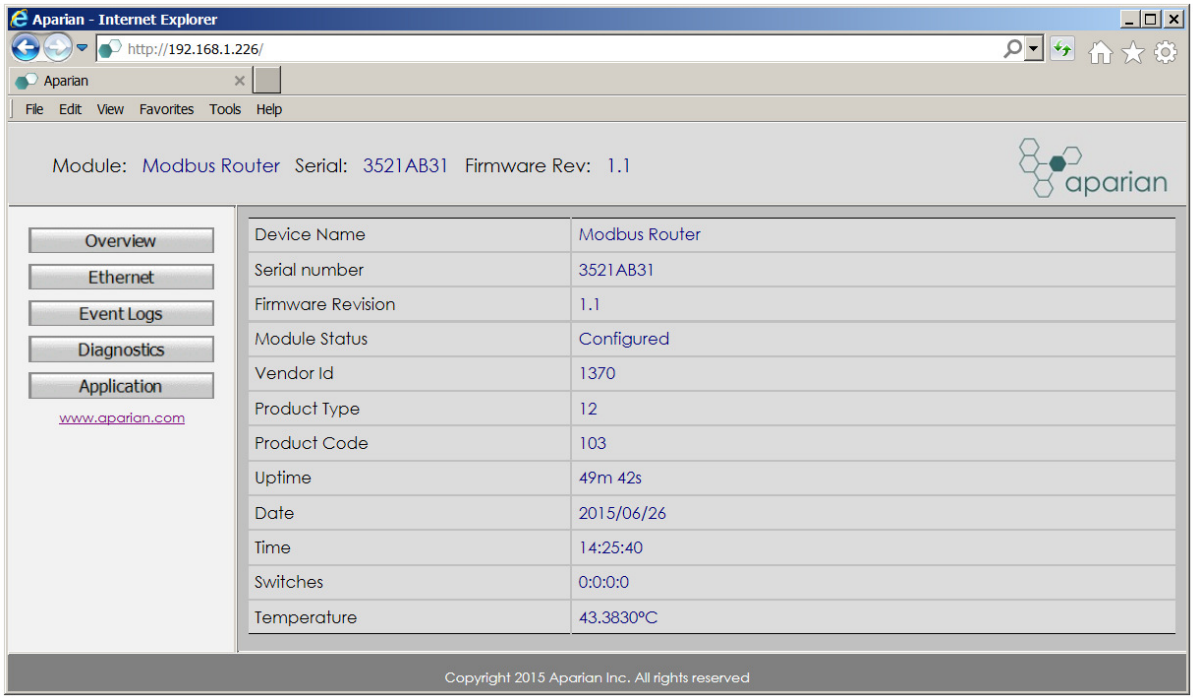

Figure 5.14. - Web interface

6. TECHNICAL SPECIFICATIONS

6.1. DIMENSIONS

Below are the enclosure dimensions as well as the required DIN rail dimensions. All dimensions are in millimetres.

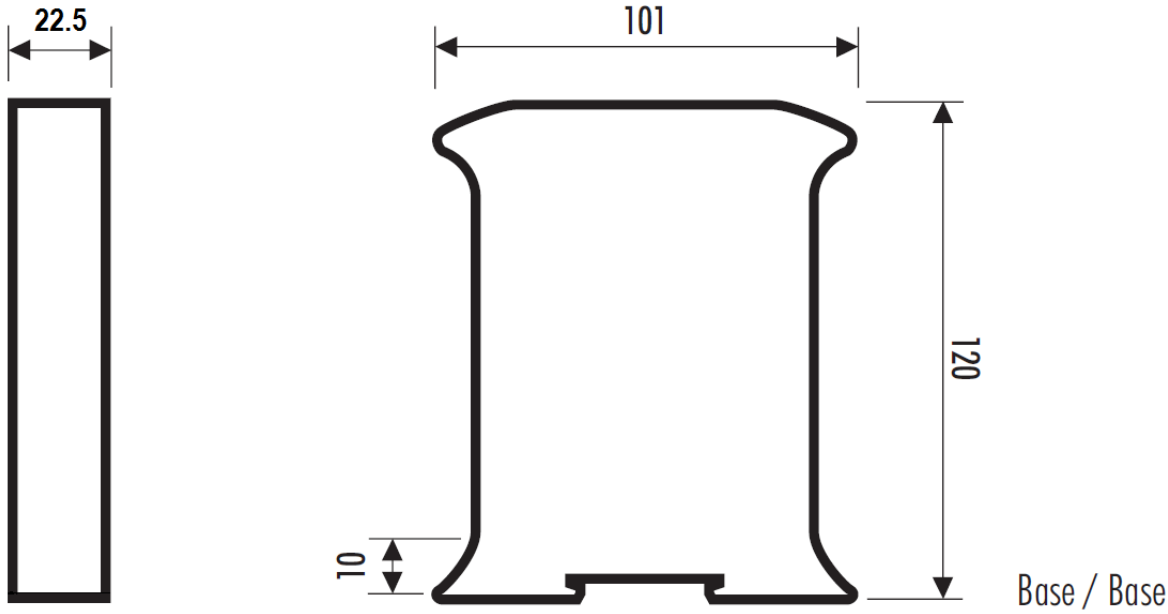

Figure 6.1 – Modbus Router enclosure dimensions

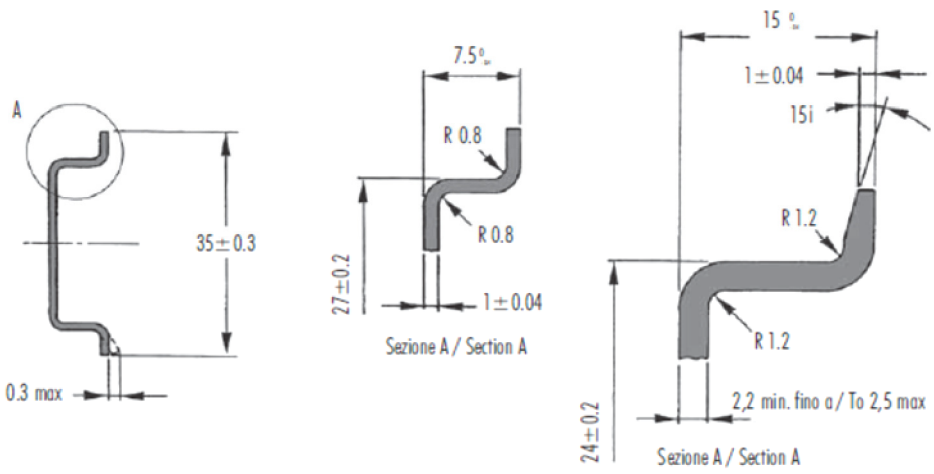

Figure 6.2 - Required DIN dimensions

6.2. ELECTRICAL

Specification	Rating
Power requirements	Input: 10 – 28V DC, (70 mA @ 24 VDC)
Power consumption	1.7 W
Connector	3-way terminal
Conductors	24 – 18 AWG
Enclosure rating	IP20, NEMA/UL Open Type
Temperature	0 – 60 °C
Earth connection	Yes, terminal based
Emissions	IEC61000-6-4
ESD Immunity	EN 61000-4-2
Radiated RF Immunity	IEC 61000-4-3
EFT/B Immunity	EFT: IEC 61000-4-4
Surge Immunity	Surge: IEC 61000-4-5
Conducted RF Immunity	IEC 61000-4-6

Table 6.1 - Electrical specification

6.3. ETHERNET

Specification	Rating
Connector	RJ45
Conductors	CAT5 STP/UTP
ARP connections	Max 20
TCP connections	Max 20
CIP connections	Max 10
Communication rate	10/100Mbps
Duplex mode	Full/Half
Auto-MDIX support	Yes

Table 6.2 - Ethernet specification

6.4. SERIAL PORT (RS232)

Specification	Rating
RS232 Connector	4-way terminal
RS232 Conductor	24 – 18 AWG
RS232 Isolation voltage	2.5 kV
BAUD	1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
Parity	None, Even, Odd
Data bits	8
Stop bits	1

Table 6.3 – RS232 Serial Port specification

6.5. SERIAL PORT (RS485)

Specification	Rating
RS485 Connector	4-way terminal
RS485 Conductor	24 – 18 AWG
BAUD	1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
Parity	None, Even, Odd
Data bits	8
Stop bits	1

Table 6.4 – RS485 Serial Port specification

6.6. MODBUS

Specification	Rating
Reactive Tag mode	Max 20 mapping items
Scheduled Tag mode	Max 20 mapping items
Application Functions Supported	Read Coil, Read Discrete Input, Read Holding Register, Read Input Register, Write Coil, Write Register
Maximum Modbus write request size	255 bytes

Maximum Logix Controller support	3
Protocols	Modbus RTU, Modbus TCP

Table 6.5 – Modbus specification

6.7. CERTIFICATIONS

Certification	Mark
CE Mark	
UL Mark File: E476538	 *

Table 6.6 – Certifications

***NOTE**: UL certification where the module label shows the UL Mark.

INDEX

U

W